パブリッククラウドでの Threat Defense Virtual のクラスタリング

クラスタリングを利用すると、複数の Firewall Threat Defense Virtual をグループ化して 1 つの論理デバイスとすることができます。クラスタは、単一デバイスのすべての利便性（管理、ネットワークへの統合）を備える一方で、複数デバイスによって高いスループットおよび冗長性を達成します。以下のパブリック クラウド プラットフォームを使用して、パブリッククラウドに Firewall Threat Defense Virtual クラスタを展開できます。

- Amazon Web Services（AWS）
- Microsoft Azure
- Google Cloud Platform（GCP）

現在は、ルーテッド ファイアウォール モードのみがサポートされます。

（注） クラスタリングを使用する場合、一部の機能はサポートされません。「サポートされていない機能とクラスタリング （122 ページ）」を参照してください。

パブリッククラウドにおける Threat Defense Virtual クラスタリングについて

ここでは、クラスタリング アーキテクチャとその動作について説明します。

クラスタをネットワークに適合させる方法

クラスタは、複数のファイアウォールで構成され、これらは 1 つのデバイスとして機能します。ファイアウォールをクラスタとして機能させるには、次のインフラストラクチャが必要です。

- クラスタ内通信用の、隔離されたネットワーク。VXLAN インターフェイスを使用したクラスタ制御リンクと呼ばれます。レイヤ 3 物理ネットワーク上でレイヤ 2 仮想ネットワークとして機能する VXLAN により、Firewall Threat Defense Virtual はクラスタ制御リンクを介してブロードキャスト/マルチキャストメッセージを送信できます。
- ロードバランサ：パブリッククラウドに応じて、外部ロードバランシングには次のオプションがあります。
  - AWS Gateway Load Balancer

    AWS ゲートウェイロードバランサは、透過的なネットワークゲートウェイと、トラフィックを分散し、仮想アプライアンスをオンデマンドで拡張するロードバランサを組み合わせます。Firewall Threat Defense Virtual は、Geneve インターフェイスのシングルアームプロキシを使用して分散データプレーン（ゲートウェイ ロード バランサ エンドポイント）を備えたゲートウェイロードバランサ集中型コントロールプレーンをサポートします。
  - Azure ゲートウェイロードバランサ

    Azure サービスチェーンでは、Firewall Threat Defense Virtual がインターネットと顧客サービス間のパケットをインターセプトできる透過的なゲートウェイとして機能します。Firewall Threat Defense Virtual は、ペアリングされたプロキシの VXLAN セグメントを利用して、単一の NIC に外部インターフェイスと内部インターフェイスを定義します。
  - 内部および外部のネイティブ GCP ロードバランサ
  - シスコ クラウド サービス ルータなどの内部および外部ルータを使用した等コスト マルチパス ルーティング（ECMP）

ECMP ルーティングでは、ルーティング メトリックが同値で最高である複数の「最適パス」を介してパケットを転送できます。EtherChannel のように、送信元および宛先の IP アドレスや送信元および宛先のポートのハッシュを使用してネクスト ホップの 1 つにパケットを送信できます。ECMP ルーティングにスタティック ルートを使用する場合は、Firewall Threat Defense の障害発生時に問題が起きることがあります。ルートは引き続き使用されるため、障害が発生した Firewall Threat Defense へのトラフィックが失われるからです。スタティック ルートを使用する場合は必ず、オブジェクト トラッキングなどのスタティック ルート モニタリング機能を使用してください。ダイナミック ルーティング プロトコルを使用してルートの追加と削除を行うことを推奨します。この場合は、ダイナミック ルーティングに参加するように各 Firewall Threat Defense を設定する必要があります。

（注） レイヤ 2 スパンド EtherChannels はロードバランシングではサポートされません。

個々のインターフェイス

クラスターフェイスを個々のインターフェイスとして設定できます。

個別インターフェイスは通常のルーテッドインターフェイスであり、それぞれが専用のローカル IP アドレスを持ちます。インターフェイスの IP アドレスは、DHCP を介して自動的に設定されます。静的 IP 設定はサポートされていません。

制御ノードとデータノードの役割

One member of the cluster is the control node. If multiple cluster nodes come online at the same time, the control node is determined by the priority setting; the priority is set between 1 and 100, where 1 is the highest priority. All other members are data nodes. When you first create the cluster, you specify which node you want to be the control node, and it will become the control node simply because it is the first node added to the cluster.

All nodes in the cluster share the same configuration. The node that you initially specify as the control node will overwrite the configuration on the data nodes when they join the cluster, so you only need to perform initial configuration on the control node before you form the cluster.

Some features do not scale in a cluster, and the control node handles all traffic for those features.

クラスタ制御リンク

Each node must dedicate one interface as a VXLAN (VTEP) interface for the cluster control link. For more information about VXLAN, see VXLAN インターフェイスの設定.

VXLAN Tunnel Endpoint

VXLAN tunnel endpoint (VTEP) devices perform VXLAN encapsulation and decapsulation. Each VTEP has two interface types: one or more virtual interfaces called VXLAN Network Identifier (VNI) interfaces, and a regular interface called the VTEP source interface that tunnels the VNI interfaces between VTEPs. The VTEP source interface is attached to the transport IP network for VTEP-to-VTEP communication.

VTEP Source Interface

The VTEP source interface is a regular Firewall Threat Defense Virtual interface with which you plan to associate the VNI interface. You can configure one VTEP source interface to act as the cluster control link. The source interface is reserved for cluster control link use only. Each VTEP source interface has an IP address on the same subnet. This subnet should be isolated from all other traffic, and should include only the cluster control link interfaces.

VNI Interface

A VNI interface is similar to a VLAN interface: it is a virtual interface that keeps network traffic separated on a given physical interface by using tagging. You can only configure one VNI interface. Each VNI interface has an IP address on the same subnet.

Peer VTEPs

Unlike regular VXLAN for data interfaces, which allows a single VTEP peer, The Firewall Threat Defense Virtual clustering allows you to configure multiple peers.

クラスタ制御リンク トラフィックの概要

Cluster control link traffic includes both control and data traffic.

Control traffic includes:

- Control node election.
- Configuration replication.
- Health monitoring.

Data traffic includes:

- State replication.
- Connection ownership queries and data packet forwarding.

コンフィギュレーションの複製

All nodes in the cluster share a single configuration. You can only make configuration changes on the control node (with the exception of the bootstrap configuration), and changes are automatically synced to all other nodes in the cluster.

管理ネットワーク

管理インターフェイスを使用して各ノードを管理する必要があります。クラスタリングでは、データインターフェイスからの管理はサポートされていません。

Threat Defense Virtual クラスタリングのライセンス

各 Firewall Threat Defense Virtual クラスタノードには、同じパフォーマンス階層ライセンスが必要です。すべてのメンバーに同じ数の CPU とメモリを使用することをお勧めします。そうしないと、パフォーマンスが最小能力のメンバーに一致するようにすべてのノードで制限されます。スループットレベルは、一致するように制御ノードから各データノードに複製されます。

個別のノードではなく、クラスタ全体に機能ライセンスを割り当てます。ただし、クラスタの各ノードは機能ごとに個別のライセンスを使用します。クラスタリング機能自体にライセンスは必要ありません。

制御ノードを Firewall Management Center に追加する際に、そのクラスタに使用する機能ライセンスを指定できます。クラスタのライセンスは、**[デバイス（Devices）]**>**[デバイス管理（Device Management）]** > **[クラスタ（Cluster）]** > **[ライセンス（License）]** 領域で変更できます。

（注） Firewall Management Center にライセンスを取得する（および評価モードで実行する）前にクラスタを追加した場合、Firewall Management Center にライセンスを取得する際にポリシーの変更をクラスタに展開するとトラフィックの中断が発生することがあります。ライセンスモードを変更したことによって、すべてのデータユニットがクラスタをいったん離れてから再参加することになります。

Threat Defense Virtual クラスタリングの要件および前提条件

モデルの要件

- FTDv5、FTDv10、FTDv20、FTDv30、FTDv50、FTDv100

（注） FTDv5 および FTDv10 は、Amazon Web Services（AWS）ゲートウェイロードバランサ（GWLB）をサポートしていません。

- 以下のパブリッククラウドサービス：
  - Amazon Web Services（AWS）

- Microsoft Azure
- Google Cloud Platform（GCP）

- 最大 16 ノード

Secure Firewall Threat Defense Virtual getting started guides の Firewall Threat Defense Virtual の一般要件も参照してください。

User roles

- Admin
- Access Admin
- Network Admin

ハードウェアおよびソフトウェアの要件

クラスタ内のすべてのユニット：

- 同じパフォーマンス層内にある必要があります。すべてのノードに同じ数の CPU とメモリを使用することをお勧めします。そうしないと、パフォーマンスが最小能力のノードに一致するようにすべてのノードで制限されます。
- Firewall Management Center へのアクセスは管理インターフェイスから行うこと。データインターフェイスの管理はサポートされていません。
- イメージアップグレード時を除き、同じソフトウェアを実行する必要があります。ヒットレス アップグレードがサポートされます。
- クラスタ内のすべてのユニットは、同じ可用性ゾーンに展開する必要があります。
- すべてのユニットのクラスタ制御リンクインターフェイスは、同じサブネット内にある必要があります。

MTU

クラスタ制御リンクに接続されているポートに適切な MTU 値（高い値）が設定されていること。MTUの不一致がある場合、クラスタの形成に失敗します。クラスタに参加したノードは、クラスタ制御リンク MTU と一致するパケット サイズで制御ノードに ping を送信することで MTU の互換性をチェックします。ping が失敗すると、通知が生成されるため、接続スイッチの MTU 不一致を修正して再試行することができます。

クラスタ制御リンクの MTU は、データインターフェイスよりも 154 バイト大きく設定されているはずです。クラスタ制御リンクのトラフィックにはデータパケット転送が含まれるため、クラスタ制御リンクはデータパケット全体のサイズに加えてクラスタトラフィックのオーバーヘッド（100 バイト）と VXLAN のオーバーヘッド（54 バイト）にも対応する必要があります。

GWLBを使用するAWSの場合、データインターフェイスはGeneveカプセル化を使用します。この場合、イーサネットデータグラム全体がカプセル化されるため、新しいパケットのサイズが大きくなるため、より大きなMTUが必要になります。送信元インターフェイスMTUをネットワークMTU＋306バイトに設定する必要があります。したがって、標準の1500 MTUネットワークパスの場合、送信元インターフェイスの MTU は 1806 であり、クラスタ制御リンクの MTU は +154 の 1960 である必要があります。

GWLB を使用する Azure の場合、データインターフェイスは VXLAN カプセル化を使用します。この場合、イーサネットデータグラム全体がカプセル化されるため、新しいパケットのサイズが大きくなるため、より大きなMTUが必要になります。クラスタ制御リンクのMTUは、送信元インターフェイスの MTU の＋80 バイトになるように設定する必要があります。

次の表は、クラスタ制御リンク MTU のデフォルト値とデータインターフェイス MTU を示しています。

（注） クラスタ制御リンクの MTU を 2561 ～ 8362 に設定することは推奨されません。ブロックプールの処理が原因で、この MTU サイズはシステム動作に最適ではありません。

表 *1*：デフォルト *MTU*

パブリック クラウド	クラスタ制御リンク MTU	データインターフェイス MTU
GWLB を使用した AWS	1980	1826
AWS	1654	1500
GWLB を使用した Azure	1454	1374
Azure	1454	1300
GCP	1554	1400

Threat Defense Virtual クラスタリングのガイドライン

ハイ アベイラビリティ

クラスタリングでは、高可用性はサポートされません。

IPv6

クラスタ制御リンクは、IPv4 のみを使用してサポートされます。

その他のガイドライン

- 大々的なトポロジ変更が発生する場合（EtherChannelインターフェイスの追加または削除、Firewall Threat Defense 上でのインターフェイスまたはスイッチの有効化または無効化、

VSS または vPC または VNet を形成するための追加スイッチの追加など）、ヘルス チェック機能や無効なインターフェイスのインターフェイス モニタリングを無効にする必要があります。トポロジの変更が完了して、コンフィギュレーション変更がすべてのユニットに同期されたら、インターフェイスのヘルス チェック機能を再度有効にできます。

- ノードを既存のクラスタに追加したときや、ノードをリロードしたときは、一時的に、限定的なパケット/接続ドロップが発生します。これは予定どおりの動作です。場合によっては、ドロップされたパケットが原因で接続がハングすることがあります。たとえば、FTP 接続の FIN/ACK パケットがドロップされると、FTP クライアントがハングします。この場合は、FTP 接続を再確立する必要があります。
- ノードでクラスタリングを無効にせずにノードの電源を切らないでください。
- 復号された TLS/SSL 接続の場合、復号状態は同期されず、接続オーナーに障害が発生すると、復号された接続がリセットされます。新規ノードへの接続を新たに確立する必要があります。復号されていない接続（復号しないルールに一致）は影響を受けず、正しく複製されます。
- ダイナミック スケーリングはサポートされていません。
- Cisco Secure Firewall バージョン 7.2 または 7.3 を使用している場合は、AWS にクラスタを展開する場合にステートフル ターゲット フェールオーバーはサポートされません。
- 各メンテナンス ウィンドウの完了後にグローバル展開を実行します。
- 自動スケール グループ（AWS）/インスタンス グループ（GCP）/スケールセット（Azure）から一度に複数のデバイスを削除しないでください。また、自動スケール グループ（AWS）/インスタンス グループ（GCP）/スケールセット（Azure）からデバイスを削除する前に、デバイスで **cluster disable** コマンドを実行することを推奨します。
- クラスタ内のデータ ノードと制御ノードを無効にする場合は、制御ノードを無効にする前にデータ ノードを無効にすることを推奨します。クラスタ内に他のデータ ノードがあるときに制御ノードが無効になっている場合は、いずれかのデータ ノードを制御ノードに昇格させる必要があります。ロールの変更はクラスタを妨害する可能性があることに注意してください。
- このガイドに記載されているカスタマイズした Day 0 構成スクリプトでは、要件に応じて IP アドレスを変更し、カスタム インターフェイス名を指定して、CCL-Link インターフェイスのシーケンスを変更することができます。
- クラウド プラットフォームに Threat Defense 仮想クラスタを展開した後の断続的な ping の失敗など、CCL が不安定になる問題が発生した場合は、CCL の不安定性の原因に対処することをお勧めします。また、CCL が不安定になる問題をある程度軽減するための一時的な回避策として、保留時間を増やすこともできます。保留時間の変更方法の詳細については、「クラスタの正常性モニタリング設定の編集」を参照してください。
- Management Center Virtual のセキュリティ ファイアウォール ルールまたはセキュリティ グループを設定する場合は、Firewall Threat Defense Virtual のプライベート IP アドレスとパブリック IP アドレスの両方を送信元 IP アドレス範囲に含める必要があります。また、Firewall Threat Defense Virtual のセキュリティ ファイアウォール ルールまたはセキュリティ

グループで、Firewall Management Center Virtual のプライベート IP アドレスとパブリック IP アドレスを指定してください。これは、クラスタリングの展開中にノードを適切に登録するために重要です。

クラスタリングのデフォルト

- cLACP システム ID は自動生成され、システムの優先順位はデフォルトでは 1 になっています。
- クラスタのヘルス チェック機能は、デフォルトで有効になり、ホールド時間は 3 秒です。デフォルトでは、すべてのインターフェイスでインターネット ヘルス モニタリングが有効になっています。
- 失敗したクラスタ制御リンクのクラスタ再結合機能が 5 分おきに無制限に試行されます。
- 失敗したデータインターフェイスのクラスタ自動再結合機能は、5 分後と、2 に設定された増加間隔で合計で 3 回試行されます。
- HTTP トラフィックでは、5 秒間の接続複製遅延がデフォルトで有効になっています。

AWS でクラスタを展開する

AWS にクラスタを展開する場合、手動で展開するか、スタックを展開する CloudFormation テンプレートを使用できます。AWS ゲートウェイロードバランサ、または Cisco Cloud Services Router などの非ネイティブのロード バランサでクラスタを使用できます。

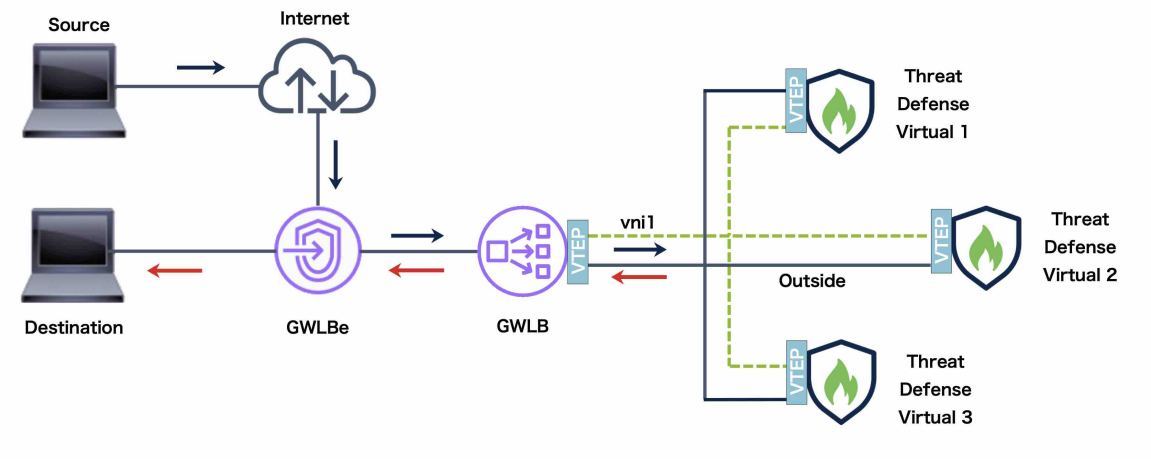
AWS ゲートウェイロードバランサおよび Geneve シングルアームプロキシ

（注） この使用例は、現在サポートされている Geneve インターフェイスの唯一の使用例です。

AWS ゲートウェイロードバランサは、透過的なネットワークゲートウェイと、トラフィックを分散し、仮想アプライアンスをオンデマンドで拡張するロードバランサを組み合わせます。Threat Defense Virtual は、分散データプレーン（ゲートウェイ ロード バランサ エンドポイント）を備えたゲートウェイロードバランサ集中型コントロールプレーンをサポートします。次の図は、ゲートウェイロードバランサのエンドポイントからゲートウェイロードバランサに転送されるトラフィックを示しています。ゲートウェイロードバランサは、複数の Threat Defense Virtual の間でトラフィックのバランスを取り、トラフィックをドロップするか、ゲートウェイロードバランサに送り返す（U ターントラフィック）前に検査します。ゲートウェイロードバランサは、トラフィックをゲートウェイロードバランサのエンドポイントと宛先に送り返します。

（注） Transport Layer Security（TLS）サーバーアイデンティティ検出は、AWS での Geneve シングルアームセットアップではサポートされていません。

図 *1 : Geneve* シングルアームプロキシ



トポロジの例

AWS リージョン内の単一および複数の可用性ゾーンにおける自動スケーリングを使用した Firewall Threat Defense Virtual クラスタリング

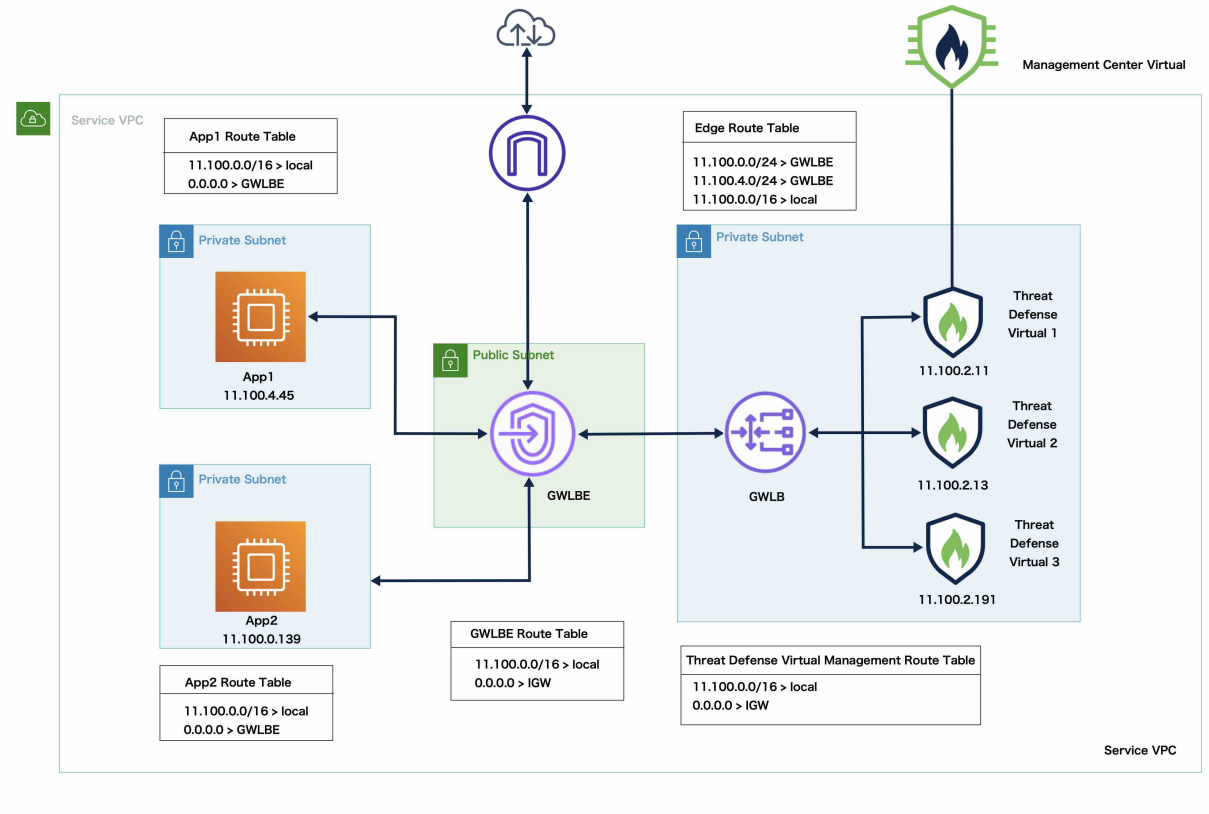
可用性ゾーンは、スタンドアロン データセンター、または独立して動作する AWS リージョン内の一連の独立したデータセンターです。各ゾーンには独自のネットワーク インフラストラクチャ、接続、および電源があり、1 つのゾーンで障害が発生しても他のゾーンには影響しません。冗長性と信頼性を向上させるために、企業は、ディザスタリカバリ計画で複数の可用性ゾーンを使用しています。

複数の可用性ゾーンに Firewall Threat Defense Virtual を展開し、動的拡張を使用してクラスタリングを設定すると、インフラストラクチャの可用性と拡張性が大幅に向上します。さらに、同じリージョン内で複数の可用性ゾーンを利用することにより、冗長性が向上し、障害発生時の高可用性が保証されます。

クラスタ制御リンク（CCL）の IP 割り当てメカニズムを変更して、AWS 上の Firewall Threat Defense Virtual クラスタについて、単一と複数の可用性ゾーンの展開を両方サポートできます。次に示すトポロジは、自動スケーリング機能を備えた単一および複数の可用性ゾーンにおける着信と発信の両方のトラフィックフローを示しています。

単一の可用性ゾーンにおける自動スケーリングを使用した Firewall Threat Defense Virtual クラスタリング

GWLB に接続されているクラスタには、2 つの Firewall Threat Defense Virtual インスタンスがあります。

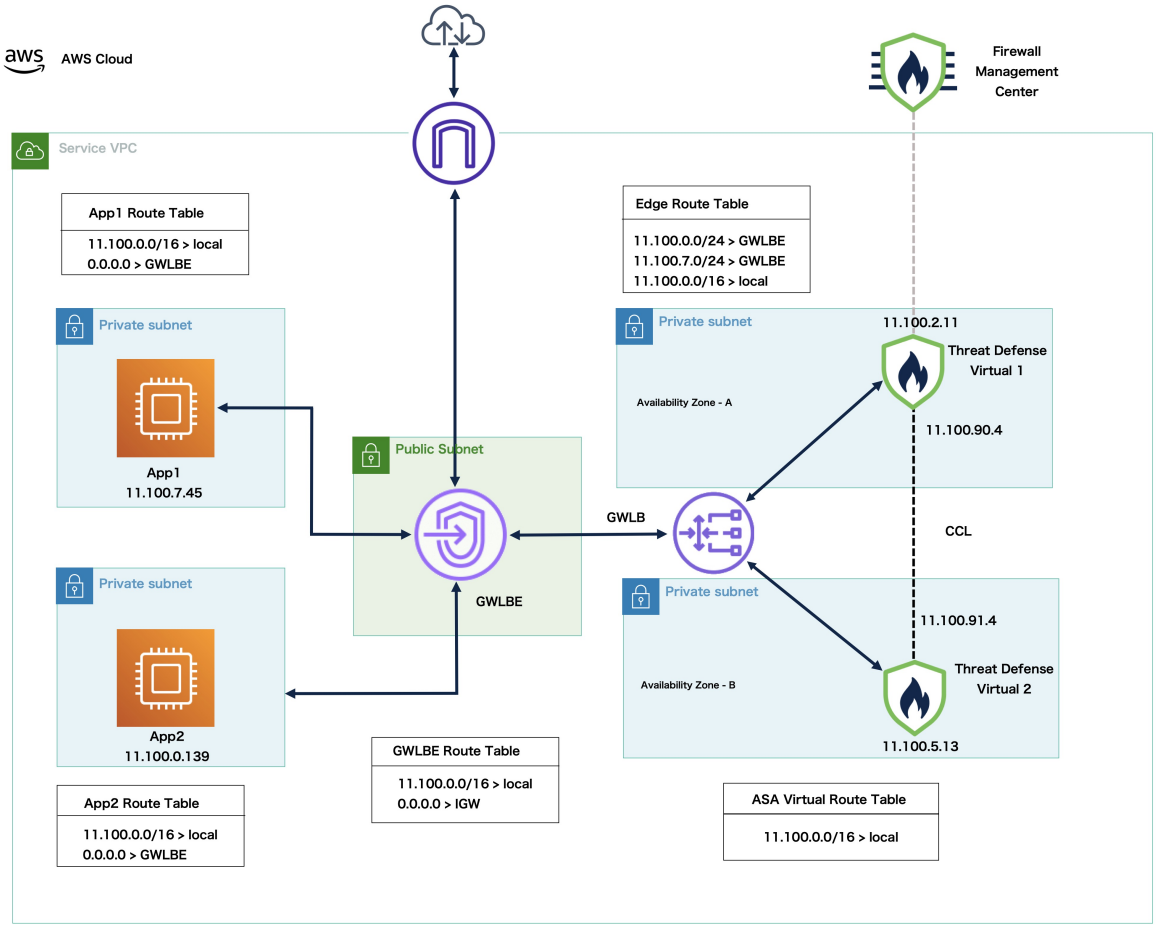


インターネットからの着信トラフィックは、GWLBエンドポイントに送られ、そこからGWLBにトラフィックが送信されます。その後、トラフィックは Firewall Threat Defense Virtual クラスタに転送されます。トラフィックは、クラスタ内の Firewall Threat Defense Virtual インスタンスによって検査された後、アプリケーション VM App1 に転送されます。

App1 からの発信トラフィックは、GWLB エンドポイント > GWLB > TDv > GWLB > GWLB エンドポイント に送信され、そこからインターネットに送信されます。

複数の可用性ゾーンにおける自動スケーリングを使用した Firewall Threat Defense Virtual クラスタリング

GWLB に接続されている異なる可用性ゾーンのクラスタには、2 つの Firewall Threat Defense Virtual インスタンスがあります。



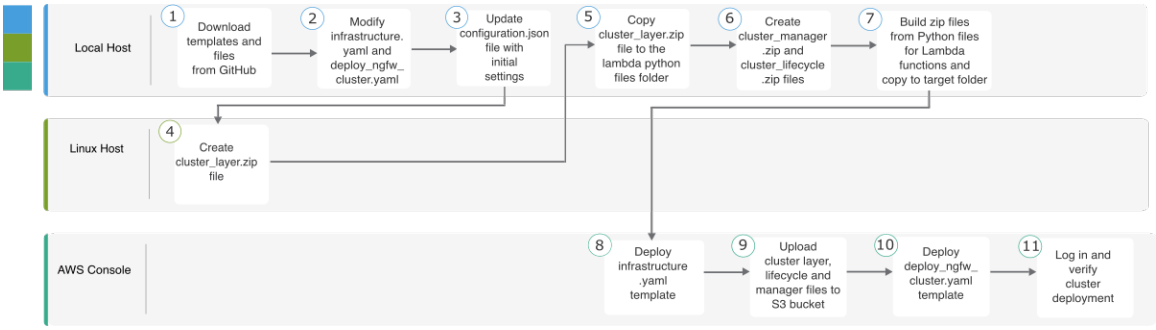
（注） 複数の可用性ゾーンの展開は、Firewall Threat Defense Virtual バージョン 7.6.0 以降でサポートされています。

インターネットからの着信トラフィックは、GWLBエンドポイントに送られ、そこからGWLBにトラフィックが送信されます。その後、可用性ゾーンに基づいて、トラフィックが Firewall Threat Defense Virtual クラスタにルーティングされます。トラフィックは、クラスタ内の Firewall Threat Defense Virtual インスタンスによって検査された後、アプリケーション VM App1 に転送されます。

AWS で Threat Defense Virtual クラスタを展開するためのエンドツーエンドのプロセス

テンプレートベースの展開

次のフローチャートは、AWS での Threat Defense Virtual クラスタのテンプレートベース展開のワークフローを示しています。



	ワークスペース	手順
1	ローカルホスト	GitHub からリポジトリを複製します。
2	ローカルホスト	infrastructure.yaml および deploy_ngfw_cluster.yaml テンプレートを変更します。
3	ローカルホスト	Configuration.json ファイルを FMC オブジェクト名で更新します。
4	Linux ホスト	cluster_layer.zip ファイルを作成します。
5	ローカルホスト	cluster_layer.zip ファイルを Lambda Python ファイルフォルダにコピーします。
6	ローカルホスト	*cluster_manager.zip*、*custom_metrics_publisher.zip*、および *cluster_lifecycle.zip* ファイルを作成します。
7	ローカルホスト	Lambda 関数の Python ファイルから zip ファイルを作成し、ターゲットフォルダにコピーします。
8	AWS コンソール	Infrastructure.yaml テンプレートを展開します。
9	AWS コンソール	*cluster_layer.zip*、*cluster_lifecycle.zip*、*custom_metrics_publisher.zip*、および *cluster_manager.zip* を S3 バケットにアップロードします。
10	AWS コンソール	deploy_ngfw_cluster.yaml テンプレートを展開します。

	ワークスペース	手順
11	AWS コンソール	ログインして、クラスタの展開を確認します。

手動展開

次のフローチャートは、AWS での Threat Defense Virtual クラスタの手動展開のワークフローを示しています。

	ワークスペース	手順
1	ローカルホスト	Day 0 構成スクリプトを作成します。
2	AWS コンソール	Threat Defense Virtual インスタンスを展開します。
3	AWS コンソール	インスタンスにインターフェイスを接続します。
4	AWS コンソール	ノードがクラスタに参加しているかどうかを確認します。
5	AWS コンソール	ターゲットグループと GWLB を作成します。ターゲットグループを GWLB に割り当てます。
6	AWS コンソール	データインターフェイス IP を使用してターゲットグループにインスタンスを登録します。
7	Management Center	制御ノードを登録します。

テンプレート

以下のテンプレートは GitHub で入手できます。パラメータ値は、テンプレートで指定されたパラメータ名、デフォルト値、使用可能な値、および説明により自明です。

- infrastructure.yaml：インフラストラクチャ展開用のテンプレート。

- deploy_ngfw_cluster.yaml：クラスタ展開用のテンプレート。

（注） クラスタノードを展開する前に、サポートされている AWS インスタンスタイプのリストを確認してください。このリストは、deploy_ngfw_cluster.yaml テンプレートのパラメータ InstanceType に使用可能な値の下にあります。

CloudFormation テンプレートを使用した AWS へのスタックの展開

CloudFormation テンプレートを使用して AWS にスタックを展開します。

始める前に

- Python 3 をインストールした Amazon Linux 仮想マシンが必要です。
- クラスタが Firewall Management Center に自動登録されるようにするには、REST API を使用できる管理者権限を持つ 2 つのユーザーを Firewall Management Center で作成する必要があります。*Cisco Secure Firewall Management Center Administration Guide*を参照してください。
- `Configuration.json` で指定したポリシー名と一致するアクセスポリシーを Firewall Management Center に追加します。

手順

ステップ 1 テンプレートを準備します。

a) GitHub リポジトリをローカルフォルダに複製します。https://github.com/CiscoDevNet/cisco-ftdv/tree/master/cluster/aws を参照してください。

b) 必要なパラメーターを使用して、**infrastructure.yaml** および **deploy_ngfw_cluster.yaml** を変更します。

c) **cluster/aws/lambda-python-files/Configuration.json** を初期設定に変更します。

次に例を示します。

```
{
  "licenseCaps": ["BASE", "MALWARE", "THREAT"],
  "performanceTier": "FTDv50",
  "fmcIpforDeviceReg": "DONTRESOLVE",
  "RegistrationId": "cisco",
  "NatId": "cisco",
  "fmcAccessPolicyName": "AWS-ACL"
}
```

- fmcIpforDeviceReg 設定は DONTRESOLVE のままにします。

- fmcAccessPolicyName は、Firewall Management Center のアクセスポリシーと一致している必要があります。

（注）
FTDv5 および FTDv10 階層はサポートされていません。

d) **cluster_layer.zip** という名前のファイルを作成して、重要な Python ライブラリを Lambda 関数に提供します。

cluster_layer.zip ファイルを作成するには、Python 3.9 がインストールされた Amazon Linux を使用することをお勧めします。

（注）
Amazon Linux 環境が必要な場合は、Amazon Linux 2023 AMI を使用して EC2 インスタンスを作成するか、Amazon Linux の最新バージョンを実行する AWS Cloudshell を使用できます。

cluster-layer.zip ファイルを作成するには、最初に Python ライブラリパッケージの詳細で構成される **requirements.txt** ファイルを作成してから、シェルスクリプトを実行する必要があります。

1. Python パッケージの詳細を指定して、**requirements.txt** ファイルを作成します。

   以下は、**requirements.txt** ファイルで指定するサンプルパッケージの詳細です。

   ```
   $ cat requirements.txt
   pycryptodome
   paramiko
   requests
   scp
   jsonschema
   cffi
   zipp
   importlib-metadata
   ```

2. 次のシェルスクリプトを実行して、**cluster_layer.zip** ファイルを作成します。

   ```
   $ pip3 install --platform manylinux2014_x86_64
   --target=./python/lib/python3.9/site-packages
   --implementation cp --python-version 3.9 --only-binary=:all:
   --upgrade -r requirements.txt
   $ zip -r cluster_layer.zip ./python
   ```

（注）
インストール中に urllib3 や暗号化などの依存関係の競合エラーが発生した場合は、競合するパッケージを推奨バージョンと一緒に **requirements.txt** ファイルに含めることをお勧めします。その後、インストールを再度実行して競合を解決できます。

e) 結果の **cluster_layer.zip** ファイルを Lambda Python ファイルフォルダ（`cluster/aws/lambda-python-files`）にコピーします。

f) **cluster_layer.zip**、**custom_metrics_publisher.zip**、**cluster_manger.zip**、および **lifecycle_ftdv.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（`cluster/aws/make.py`）内にあります。これにより、python ファイルが Zip ファイルに圧縮され、ターゲットフォルダにコピーされます。

python3 make.py build

（注）
Management Center Virtual の登録にプライベート IP アドレスを使用している場合は、`cisco-ftdv/cluster/aws/lambda-python-files/constant.py` ファイルで USE_PUBLIC_IP_FOR_FMC_CONN を False に設定していることを確認します。

ステップ 2 **Infrastructure.yaml** を展開し、クラスタ展開の出力値をメモします。インフラストラクチャ スタックを展開する前に、使用する AWS リージョンと可用性ゾーンを特定することが重要です。各 AWS リージョンには異なる可用性ゾーンと VPC インフラストラクチャのセットがあるため、展開に適したリージョンと可用性ゾーンを選択することが不可欠です。

a) AWS コンソールで、[CloudFormation] に移動し、[新しいリソース（標準）を使用（With new resources(standard)）] を選択して、[スタックの作成（Create stack）] をクリックします。
b) [テンプレートファイルのアップロード（Upload a template file）] を選択し、[ファイルの選択（Choose file）] をクリックして、ターゲットフォルダから **infrastructure.yaml** を選択します。
c) [次へ（Next）] をクリックして、必要な情報を入力します。
d) クラスタの一意の**クラスタ名**と**クラスタ番号**を入力します。
e) [可用性ゾーン（Availability Zone）] リストから可用性ゾーンを選択します。このフィールドには、ClusterFormation テンプレートを使用してインフラストラクチャ スタックを展開するために選択した AWS リージョンに基づく可用性ゾーンのみが表示されます。
f) [次へ（Next）]、[スタックの作成（Create stack）] の順にクリックします。
g) 展開が完了したら、[出力（Outputs）] に移動し、S3 の **BucketName** を書き留めます。

図 *2 : Infrastructure.yaml* の出力

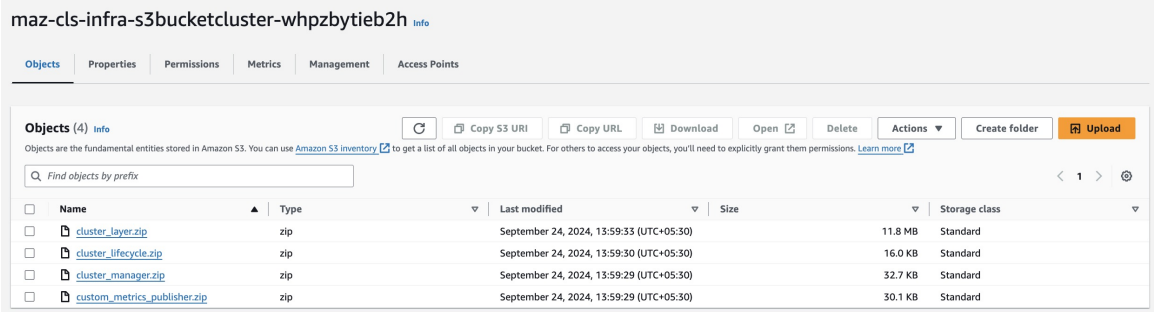
Outputs (13)

Search outputs

Key	Value	Description	Export name
BucketName	maz-cls-infra-s3bucketcluster-whpzbytieb2h	Name of the Amazon S3 bucket	-
BucketUrl	http://maz-cls-infra-s3bucketcluster-whpzbytieb2h.s3-website-us-east-1.amazonaws.com	URL of S3 Bucket Static Website	-
CCLSubnetIds	subnet-0bc04e2cc9e53e5c0,subnet-0d7d046a0fca25615,subnet-03ef42bf527551569	List of CCL subnet IDs (comma seperated)	-
EIPforNATgw	3.218.44.132	EIP reserved for NAT GW	-
FmcInstanceSGID	sg-076880aa64df2db5c	Security Group ID for FMC if user would like to launch in this VPC itself	-
InInterfaceSGId	sg-06ed933d6624fe51b	Security Group ID for Inside Interfaces	-
InsideSubnetIds	subnet-03d12cab8ee0eafff,subnet-0be9158b0970aebab,subnet-0b53c96fceb7c1f4d	List of Inside subnet IDs (comma seperated)	-
InstanceSGId	sg-0680b74be473186aa	Security Group ID for Instances Management Interface	-
LambdaSecurityGroupId	sg-057da2a9954e0d204	Security Group ID for Lambda Functions	-
LambdaSubnetIds	subnet-03439803d989e6bdf,subnet-087488a9d6ffc95cd	List of lambda subnet IDs (comma seperated)	-
ListOfAZs	us-east-1a,us-east-1b,us-east-1c	Availability zones for NGFWv instances	-
MgmtSubnetIds	subnet-06f0bbbd3f207a504,subnet-0c339dc43688cddc9,subnet-0a67629632a655de7	List of Mangement subnet IDs (comma seperated)	-
VpcName	vpc-09c2b0ad995e2fb24	Name of the VPC created	-

ステップ 3 **cluster_layer.zip**、**cluster_manager.zip**、**custom_metrics_publisher.zip**、および **cluster_lifecycle.zip** を **infrastructure.yaml** で作成した S3 バケットにアップロードします。

図 *3 : S3* バケット



（注）
Lambda NAT ゲートウェイのエラスチック IP アドレスが Management Center Virtual に関連付けられたセキュリティグループに追加されていることを確認してください。

ステップ 4 **deploy_ngfw_cluster.yaml** を展開します。

a) [CloudFormation] に移動し、[新しいリソース（標準）を使用（With new resources(standard)）] を選択して、[スタックの作成（Create stack）] をクリックします。

b) [テンプレートファイルのアップロード（Upload a template file）] を選択し、[ファイルの選択（Choose file）] をクリックして、ターゲットフォルダから **deploy_ngfw_cluster.yaml** を選択します。

c) [次へ（Next）] をクリックして、必要な情報を入力します。

d) 次のクラスタとインフラストラクチャの設定情報を入力します。

パラメータ	使用できる値/タイプ	説明
クラスタの設定		
ClusterGrpNamePrefix	文字列	これはクラスタ名のプレフィックスです。クラスタ番号がサフィックスとして追加されます。
ClusterNumber	文字列	これはクラスタ番号であり、クラスタ名（msa-ftdv-infra）のサフィックスとして追加されます。たとえば、この値が「1」の場合、グループ名は msa-ftdv-infra-1 になります。 1 桁以上 3 桁以下である必要があります。デフォルト：1。
ClusterSize	番号	これは、クラスタ内の Firewall Threat Defense Virtual ノードの総数です。 最小値：1

パラメータ	使用できる値/タイプ	説明
		最大値：16
インフラストラクチャの詳細		
NoOfAZs	文字列	これは、Firewall Threat Defense Virtual が展開される可用性ゾーンの合計数です（可用性ゾーンの数は 1 ～ 3 でリージョンによって異なります）。 サブネットは、これらの可用性ゾーンに作成されます。 このリストで使用可能な可用性ゾーンは、クラスタの展開用に選択されたリージョンに基づいています。 （注） 管理、内部、およびクラスタ制御リンク（CCL）のサブネットは、このパラメータに基づいて 3 つの可用性ゾーンにわたって作成されます。
AZ	文字列	可用性ゾーンリストは、展開するリージョンに基づきます。 [可用性ゾーン（Availability Zone）] リストで、有効な可用性ゾーン（1 つの可用性ゾーン、2 つの可用性ゾーン、または 3 つの可用性ゾーン）を選択します。 カウントは、可用性ゾーン数のパラメータの値と一致する必要があります。
NotifyEmailID	文字列	クラスタイベントの電子メールの送信先となる電子メールアドレス。この電子メール通告を受信するには、サブスクリプション電子メール要求を承認する必要があります。 例：admin@company.com
VpcId	文字列	クラスタグループの VPC ID。 タイプ：AWS::EC2::VPC::Id
S3BktName	文字列	アップロードされた Lambda zip ファイルを含む S3 バケット。正しいバケット名を指定する必要があります。
MgmtSubnetIds	リスト	可用性ゾーンごとに「1 つ」の サブネットのみを入力します。 同じ可用性ゾーンから複数のサブネットを選択する場合、間違ったサブネットを選択すると、Firewall Threat

パラメータ	使用できる値/タイプ	説明
		Defense Virtual インスタンスの展開中に問題が発生する可能性があります。 タイプ：List<AWS::EC2::Subnet::Id>
InsideSubnetIds	リスト	可用性ゾーンごとに少なくとも「1 つ」の サブネットを入力します。 同じ可用性ゾーンから複数のサブネットを選択する場合、間違ったサブネットを選択すると、Firewall Threat Defense Virtual インスタンスの展開中に問題が発生する可能性があります。 タイプ：List<AWS::EC2::Subnet::Id>
LambdaSubnets	リスト	Lambda 関数用に少なくとも「2 つ」のサブネットを入力します。Lambda 関数が、パブリック DNS である AWS サービスと通信できるようにするには、入力する「2つ」のサブネットにNAT ゲートウェイが必要です。 タイプ：List<AWS::EC2::Subnet::Id>
CCLSubnetIds	文字列	可用性ゾーンごとに少なくとも「1 つ」の サブネットを入力します。 同じ可用性ゾーンから複数のサブネットを選択する場合、間違ったサブネットを選択すると、Firewall Threat Defense Virtual インスタンスの展開中に問題が発生する可能性があります。 タイプ：List<AWS::EC2::Subnet::Id>
CCLSubnetRanges	文字列	さまざまな可用性ゾーンの CCL サブネットの IP アドレス範囲を入力します。 最初の 4 つの予約済み IP アドレスを除外します。クラスタ制御リンク（CCL）の IP アドレスプール。 IP アドレスは、CCL IP アドレスプールから Firewall Threat Defense Virtual インスタンスの CCL インターフェイスに割り当てられます。
MgmtInterfaceSG	リスト	Firewall Threat Defense Virtual インスタンスのセキュリティグループ ID を選択します。 タイプ：List<AWS::EC2::SecurityGroup::Id>

パラメータ	使用できる値/タイプ	説明
InsideInterfaceSG	リスト	Firewall Threat Defense Virtual インスタンスの内部インターフェイスのセキュリティグループ ID を選択します。 タイプ：List<AWS::EC2::SecurityGroup::Id>
LambdaSG	リスト	Lambda 関数のセキュリティグループを選択します。 発信接続が [ANYWHERE] に設定されていることを確認します。 タイプ：List<AWS::EC2::SecurityGroup::Id>
CCLInterfaceSG	リスト	Firewall Threat Defense Virtual インスタンスの CCL インターフェイスのセキュリティグループ ID を選択します。
GWLB の設定		
DeployGWLBE	文字列	[はい（Yes）] をクリックして GWLB エンドポイントを展開します。 デフォルトでは、この値は [いいえ（No）] に設定されています。
VpcIdLBE	文字列	ゲートウェイロードバランサのエンドポイントを展開する VPC を入力します。 （注） GWLBエンドポイントを展開しない場合は、このフィールドに値を入力しないでください。
GWLBESubnetId	文字列	サブネット ID を 1 つだけ入力します。 （注） GWLBエンドポイントを展開しない場合は、このフィールドに値を入力しないでください。 サブネットが正しい VPC および指定した可用性ゾーンに属していることを確認します。
TargetFailover	文字列	ターゲットに障害が発生した場合または登録解除された場合のターゲット フェールオーバー サポートを有効にします（このパラメータの値はデフォルトで [再調整（rebalance）] に設定されています）。

パラメータ	使用できる値/タイプ	説明
		• [再調整なし（no_rebalance）]：既存のフローを障害が発生したターゲットに送信し、新しいフローを正常なターゲットに送信し、後方互換性を確保します。 • [再調整（rebalance）]：新しいフローが正常なターゲットに送られるようにしながら、既存のフローを再配布します。 [再調整（rebalance）] は、Firewall Threat Defense Virtual バージョン 7.4.1 以降でサポートされています。
TgHealthPort	文字列	GWLB の正常性チェックポートを入力します。 （注） デフォルトでは、このポートはトラフィックに使用されません。 指定した値が有効な TCP ポートであることを確認します。デフォルト：8080
Cisco NGFWv インスタンスの設定		
InstanceType	文字列	Cisco Firewall Threat Defense Virtual EC2 インスタンスタイプ。 選択したインスタンスタイプが AWS リージョンでサポートされていることを確認します。 デフォルトでは、**c5.xlarge** が選択されています。
LicenseType	文字列	Cisco Firewall Threat Defense Virtual EC2 インスタンス ライセンス タイプを選択します。**AMI-ID** パラメータに入力する AMI ID が同じライセンスタイプであることを確認します。 デフォルトでは、[BYOL] が選択されています。
AssignPublicIP	文字列	AWS IP アドレスプールから Firewall Threat Defense Virtual のパブリック IP アドレスを割り当てるには、値を [はい（true）] に設定します。
AmiID	文字列	リージョン、バージョン、およびライセンスタイプ（BYOL または PAYG）に従って正しい AMI ID を選択します。

パラメータ	使用できる値/タイプ	説明
		Firewall Threat Defense Virtual 7.2 以降ではクラスタリングがサポートされ、Firewall Threat Defense Virtual バージョン 7.6 以降では自動スケーリングと複数の可用性ゾーンの機能拡張がサポートされています。 タイプ：AWS::EC2::Image::Id
ngfwPassword	文字列	Firewall Threat Defense Virtual インスタンスのパスワード。 すべての Firewall Threat Defense Virtual インスタンスには、起動テンプレート（クラスタグループ）の [ユーザーデータ（Userdata）] フィールドにあるデフォルトのパスワードが設定されています。 Firewall Threat Defense Virtual にアクセスできるようになると、パスワードがアクティブになります。 文字数は 8 文字以上にする必要があります。パスワードには、プレーンテキストのパスワードまたはKMS暗号化パスワードを使用できます。
KmsArn	文字列	既存の KMS の ARN（保存時に暗号化するための AWS KMS キー）を入力します。 このフィールドに値を指定する場合、Firewall Threat Defense Virtual インスタンスの**管理者**パスワードは暗号化されたパスワードである必要があります。 暗号化パスワードの生成例："*aws kms encrypt --key-id <KMS ARN> --plaintext <password>* " パスワードの暗号化は、指定されたARNのみを使用して実行する必要があります。
FMC 自動化の設定		
fmcDeviceGrpName	文字列	Management Center でクラスタグループの一意の名前を入力します。
fmcPublishMetrics	文字列	Management Center をポーリングし、特定のデバイスグループメトリックを AWS CloudWatch にパブリッシュする Lambda 関数を作成するには、**`true`** を選択します。 使用可能な値： • true

パラメータ	使用できる値/タイプ	説明
		• false デフォルトでは、この値は **true** に設定されています。
fmcMetricsUsername	文字列	Management Center からメモリメトリックをポーリングするための一意の内部ユーザー名を入力します。 ユーザーには、**ネットワーク管理者**および**メンテナンスユーザー**以上の権限が必要です。
fmcMetricsPassword	文字列	パスワードを入力します。 KMS マスターキー ARN パラメータを指定した場合は、必ず、暗号化されたパスワードを入力してください。 間違ったパスワードを入力するとメトリック収集が失敗する可能性があるため、必ず、正しいパスワードを入力してください。
fmcServer	文字列	IP アドレスには、外部 IP アドレス、または VPC の Firewall Threat Defense Virtual 管理サブネットで到達可能な IP アドレスを指定できます。 最小長：7 最大長：15
fmcOperationsUsername	文字列	CloudWatch 用の Firewall Management Center Virtual に使用する一意の内部ユーザー名を入力します。 ユーザーには**管理者**権限が必要です。
fmcOperationsPassword	文字列	パスワードを入力します。 KMS マスターキー ARN パラメータを指定した場合は、必ず、暗号化されたパスワードを入力してください。
スケーリングの設定		
CpuThresholds	カンマ区切りリスト	（任意）ゼロ以外の下限しきい値と上限しきい値を指定すると、スケールポリシーが作成されます。(0,0) を選択すると、CPU スケーリングアラームまたはポリシーは作成されません。評価ポイントとデータポイントは、デフォルト値または推奨値にします。 デフォルトでは、このテンプレートでは**自動スケール**が有効になっています。自動スケールは展開後に無効にできます。

パラメータ	使用できる値/タイプ	説明
MemoryThresholds	カンマ区切りリスト	ゼロ以外の下限しきい値と上限しきい値を指定すると、スケールポリシーが作成されます。(0,0) を選択すると、メモリスケーリングアラームまたはポリシーは作成されません。評価ポイントとデータポイントは、デフォルト値または推奨値にします。

e) [次へ（Next）] をクリックします。
f) クリックしてすべての AWS CloudFormation オプションを確認します。
g) [送信（Submit）] をクリックしてクラスタを展開します。
h) [次へ（Next）]、[スタックの作成（Create stack）] の順にクリックします。

Lambda 関数が残りのプロセスを管理し、Firewall Threat Defense Virtual が自動的に Firewall Management Center に登録されます。

図 *4:* 展開されたリソース

ステータスが **CREATE_IN_PROGRESS** から **CREATE COMPLETE** に変わり、展開が成功したことが示されます。

ステップ 5 いずれかのノードにログインし、**show cluster info** コマンドを使用して、クラスタの展開を確認します。

図 *5:* クラスタ ノード

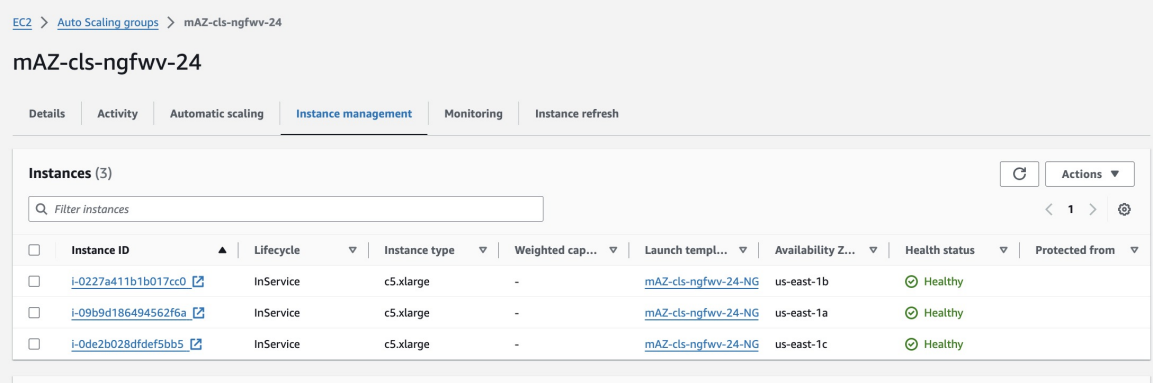
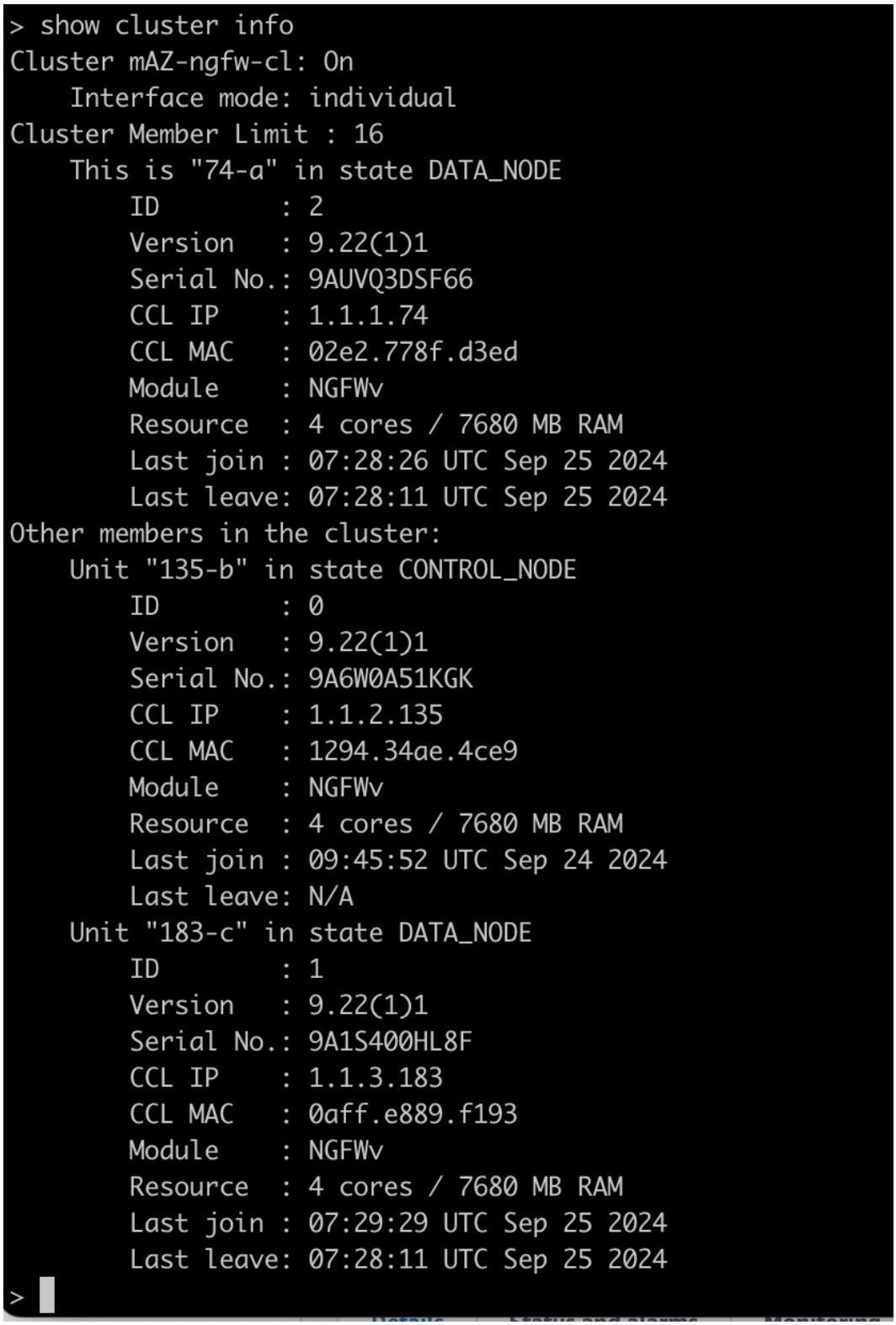


図 *6 : show cluster info*

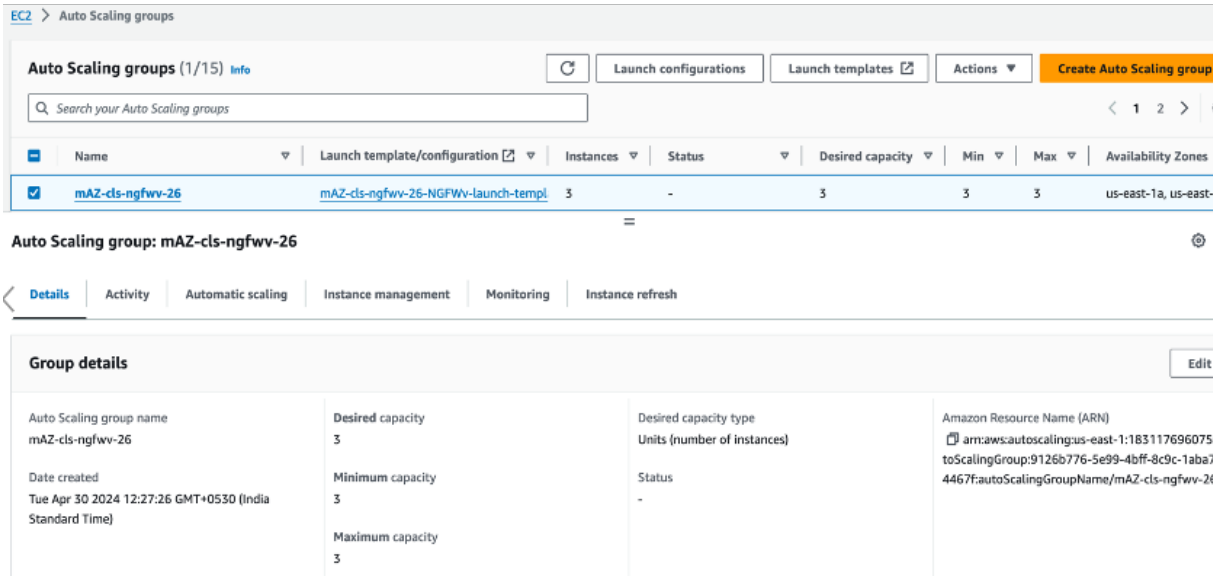


自動スケーリングのパラメータ設定

展開が完了したら、Firewall Threat Defense Virtual 自動スケーリンググループの [最小容量（Minimum capacity）]、[最大容量（Maximum capacity）]、および [必要な容量（Desired capacity）] を指定する必要があります。自動スケーリング機能を検証する必要があります。

手順

ステップ 1 AWS コンソールから、**[サービス（Services）]** > **[EC2]** > **[自動スケーリンググループ（Auto Scaling groups）]** > **作成された ClusterAutoscale グループ**の順に選択します。



ステップ 2 [自動スケーリンググループ（autoscale group）] チェックボックスをオンにします。

ステップ 3 [アクション（Actions）] をクリックして、自動スケーリンググループの容量を編集します。

ステップ 4 [必要な容量（Desired capacity）] を設定してから、[スケーリング制限（Scaling limits）] 容量を設定します。

ステップ 5 CPU とメモリのメトリックデータが使用可能かどうかと、AWS Cloudwatch アラームでスケーリングが予期したとおりに発生しているかどうかを確認します。

スタックの更新による Firewall Threat Defense Virtual クラスタリングでの IMDSv2 必須モードの設定

AWS にすでに展開されている Firewall Threat Defense Virtual の自動スケール グループ インスタンスの IMDSv2 必須モードを設定できます。

Before you begin

IMDSv2 必須モードは、Firewall Threat Defense Virtual バージョン 7.6 以降でのみサポートされています。展開にIMDSv2モードを設定する前に、既存のインスタンスのバージョンがIMDSv2モードと互換性がある（バージョン 7.6 にアップグレードされている）ことを確認する必要があります。

Procedure

ステップ 1 AWS コンソールで、[CloudFormation] に移動し、[スタック（Stacks）] をクリックします。
ステップ 2 最初に展開されたクラスタリングインスタンスのスタックを選択します。
ステップ 3 **[更新（Update）]**をクリックします。
ステップ 4 [スタックの更新（Update stack）] ページで、[既存のテンプレートの置換（Replace existing template）] をクリックします。
ステップ 5 [テンプレートの指定（Specify template）] セクションで、[テンプレートファイルのアップロード（Upload a template file）] をクリックします。
ステップ 6 IMDSv2 をサポートするテンプレートを選択してアップロードします。
ステップ 7 テンプレートの入力パラメータの値を指定します。
ステップ 8 スタックを更新します。

AWS でのクラスタの手動展開

クラスタを手動で展開するには、Day 0 構成を準備し、各ノードを展開してから制御ノードを Firewall Management Center に追加します。

AWS 向け Day 0 構成の作成

固定構成またはカスタマイズ構成のいずれかを使用できます。固定構成の使用をお勧めします。

AWS 向け固定構成を使用した Day 0 構成の作成

固定構成により、クラスタのブートストラップ構成が自動生成されます。

単一の可用性ゾーン：AWS 向け固定構成を使用した Day 0 構成

```
{
    "AdminPassword": "password",
    "Hostname": "hostname",
    "FirewallMode": "Routed",
    "ManageLocally": "No",
    "Cluster": {
        "CclSubnetRange": "ip_address_start ip_address_end",
        "ClusterGroupName": "cluster_name",
        [For Gateway Load Balancer] "Geneve": "{Yes | No}",
```

```
        [For Gateway Load Balancer] "HealthProbePort": "port"
    }
}
```

次に例を示します。

```
{
 "AdminPassword": "Sup3rnatural",
 "Hostname": "ciscoftdv",
 "FirewallMode": "Routed",
 "ManageLocally": "No",
 "Cluster": {
              "CclSubnetRange": "10.5.90.4 10.5.90.30",
  "ClusterGroupName": "ftdv-cluster",
  "Geneve": "Yes",
  "HealthProbePort": "7777"
 }
}
```

複数の可用性ゾーン：AWS 向け固定構成を使用した Day 0 構成

```
{
    "AdminPassword": "password",
    "Hostname": "hostname",
    "FirewallMode": "Routed",
    "ManageLocally": "No",
    "Cluster": {
    "CclSubnetRange":[
    "ip_address_start_AZ1 ip_address_end_AZ1",
    "ip_address_start_AZ2 ip_address_end_AZ2",
    "ip_address_start_AZ3 ip_address_end_AZ3"
     ],
   "ClusterGroupName": "cluster_name",
        [For Gateway Load Balancer] "Geneve": "{Yes | No}",
        [For Gateway Load Balancer] "HealthProbePort": "port"
    }
}
```

例：2 つの可用性ゾーン

```
{
 "AdminPassword": "Sup4rnatural",
 "Hostname": "ftdvcluster",
 "FirewallMode": "Routed",
 "ManageLocally": "No",
 "Cluster": {
        "CclSubnetRange": [
            "10.5.90.4  10.5.90.30",
            "10.5.91.4  10.5.91.30"
        ],
  "ClusterGroupName": "ftdv-cluster",
  "Geneve": "Yes",
  "HealthProbePort": "8080"
 }
}
```

例：3 つの可用性ゾーン

```
{
 "AdminPassword": "Sup4rnatural",
```

```
 "Hostname": "ftdvcluster",
 "FirewallMode": "Routed",
 "ManageLocally": "No",
 "Cluster": {
        "CclSubnetRange": [
            "10.5.90.4  10.5.90.30",
            "10.5.91.4  10.5.91.30",
            "10.5.92.4  10.5.92.30"
        ],
  "ClusterGroupName": "ftdv-cluster",
  "Geneve": "Yes",
  "HealthProbePort": "8080"
 }
}
```

CclSubnetRange 変数には、x.x.x.4 から始まる IP アドレスの範囲を指定します。クラスタリングに使用可能な IP アドレスが 16 個以上あることを確認します。開始（`ip_address_start`）および終了（`ip_address_end`）IP アドレスの例を以下に示します。

表 *2:* 開始 *IP* アドレスと終了 *IP* アドレスの例

CIDR	開始 IP アドレス	終了 IP アドレス
10.1.1.0/27	10.1.1.4	10.1.1.30
10.1.1.32/27	10.1.1.36	10.1.1.62
10.1.1.64/27	10.1.1.68	10.1.1.94
10.1.1.96/27	10.1.1.100	10.1.1.126
10.1.1.128/27	10.1.1.132	10.1.1.158
10.1.1.160/27	10.1.1.164	10.1.1.190
10.1.1.192/27	10.1.1.196	10.1.1.222
10.1.1.224/27	10.1.1.228	10.1.1.254
10.1.1.0/24	10.1.1.4	10.1.1.254

クラスタノードの展開

クラスタが形成されるようにクラスタノードを展開します。

手順

ステップ 1 必要な数のインターフェイス（ゲートウェイロードバランサ（GWLB）を使用している場合は 4 つのインターフェイス、非ネイティブロードバランサを使用している場合は 5 つのインターフェイス）でクラスタの Day 0 構成を使用することにより、Threat Defense Virtual インスタンスを展開します。これを行うには、[インスタンスの詳細設定（Configure Instance Details）] > [高度な詳細（Advanced Details）] セクションで、クラスタの Day 0 構成に貼り付けます。

（注）

次の順序でインスタンスにインターフェイスを接続していることを確認します。

- AWS ゲートウェイロードバランサの 4 つのインターフェイス：管理、診断、内部、クラスタ制御リンク。
- 非ネイティブロードバランサの 5 つのインターフェイス：管理、診断、内部、外部、クラスタ制御リンク。

AWS での Threat Defense Virtual の展開の詳細については、「Deploy the Threat Defense Virtual on AWS」を参照してください。

ステップ 2 ステップ 1 を繰り返して、必要な数の追加ノードを展開します。

ステップ 3 Threat Defense Virtual コンソールで **show cluster info** コマンドを使用して、すべてのノードがクラスタに正常に参加したかどうかを確認します。

ステップ 4 AWS ゲートウェイロードバランサを設定します。

a) ターゲットグループと GWLB を作成します。

b) ターゲットグループを GWLB に割り当てます。

（注）
正しいセキュリティグループ、リスナー設定、およびヘルスチェック設定を使用するように GWLB を設定していることを確認します。

c) IP アドレスを使用して、データインターフェイス（内部インターフェイス）をターゲットグループに登録します。

詳細については、「Create a Gateway Load Balancer」を参照してください。

ステップ 5 Management Center に制御ノードを追加します。「Management Center へのクラスタの追加（手動展開） （96 ページ）」を参照してください。

AWS における GWLB を使用した Secure Firewall Threat Defense Virtual クラスタリングのターゲットフェールオーバーの設定

AWS の Threat Defense 仮想クラスタリングは、ゲートウェイロードバランサ（GWLB）を使用して、指定された Threat Defense 仮想ノードにインスペクション用のネットワークパケットをバランシングおよび転送します。GWLB は、ターゲットノードのフェールオーバーまたは登録解除が発生した場合に、ターゲットノードにネットワークパケットを送信し続けるように設計されています。

AWS でターゲットフェールオーバー機能を使用すると、計画メンテナンス中またはターゲットノードの障害時にノードの登録が解除された場合に、GWLB がネットワークパケットを正常なターゲットノードにリダイレクトできるようになります。この機能ではクラスタのステートフル フェールオーバーを利用しています。

AWS では、AWS Elastic Load Balancing（ELB）API または AWS コンソールを介してターゲットフェールオーバーを設定できます。

（注） GWLB が SSH、SCP、CURL、などの特定のプロトコルを使用してトラフィックをルーティングしている間にターゲットノードに障害が発生した場合、正常なターゲットへのトラフィックのリダイレクトに遅延が発生する可能性があります。この遅延は、トラフィックフローの再調整と再ルーティングが原因で発生します。

AWS では、AWS ELB API または AWS コンソールを介してターゲットフェールオーバーを設定できます。

- AWS API（AWS ELB API 内）の *modify-target-group-attributes* で、次の 2 つの新しいパラメータを変更することにより、フロー処理の動作を定義できます。
  - target_failover.on_unhealthy：ターゲットが正常でなくなった場合に GWLB がネットワークフローをどのように処理するかを定義します。
  - target_failover.on_deregistration：ターゲットが登録解除された場合に GWLB がネットワークフローをどのように処理するかを定義します。

  次のコマンドは、これら 2 つのパラメータを定義するサンプルの API パラメータ設定を示しています。

```
aws elbv2 modify-target-group-attributes \
--target-group-arn arn:aws:elasticloadbalancing:…/my-targets/73e2d6bc24d8a067 \
--attributes \
Key=target_failover.on_unhealthy, Value=rebalance[no_rebalance] \
Key=target_failover.on_deregistration, Value=rebalance[no_rebalance]
```

  詳細については、AWS のマニュアルの「TargetGroupAttribute」を参照してください。

- AWS コンソール：EC2 コンソールでは、次のオプションを設定することにより、[ターゲットグループ（Target Group）]ページの[ターゲットフェールオーバー（Target Failover）]オプションを有効にすることができます。
  - ターゲットグループの属性を編集する
  - ターゲットフェールオーバーを有効にする
  - 再調整フローを確認する

  ターゲットフェールオーバーを有効にする方法の詳細については、AWS における Secure Firewall Threat Defense Virtual クラスタリングのターゲットフェールオーバーの有効化（34 ページ） を参照してください。

AWS における Secure Firewall Threat Defense Virtual クラスタリングのターゲットフェールオーバーの有効化

Firewall Threat Defense Virtual のデータインターフェイスは、AWS の GWLB のターゲットグループに登録されます。Firewall Threat Defense Virtual クラスタリングでは、各インスタンスはターゲットグループに関連付けられています。GWLB は、ターゲットグループのターゲット

ノードとして識別または登録されているこの正常なインスタンスにトラフィックを負荷分散して送信します。

始める前に

手動で、または CloudFormation テンプレートを使用して、AWS でクラスタを展開している必要があります。

CloudFormation テンプレートを使用してクラスタを展開する場合、クラスタ展開ファイル `deploy_ftdv_clustering.yaml` の [GWLBの設定（GWLB Configuration）] セクションで使用可能な [再調整（rebalance）] 属性を割り当てることにより、[ターゲットフェールオーバー（Target Failover）] のパラメータを有効にすることもできます。テンプレートでは、このパラメータの値はデフォルトで [再調整（rebalance）] に設定されています。ただし、AWS コンソールでは、このパラメータのデフォルト値は [再調整なし（no_rebalance）] に設定されています。

それぞれの説明は次のとおりです。

- 再調整なし（no_rebalance）：GWLB は、機能していないターゲットまたは登録解除されたターゲットにネットワークフローを送信し続けます。
- **再調整（rebalance）**：既存のターゲットが機能していないか登録解除された場合、GWLB はネットワークフローを別の正常なターゲットに送信します。

AWS でのスタックの展開については、以下を参照してください。

- AWS でのクラスタの手動展開
- CloudFormation テンプレートを使用した AWS へのスタックの展開

手順

ステップ 1 AWS コンソールで、**[サービス（Services）] > [EC2]** に移動します。

ステップ 2 [ターゲットグループ（Target Groups）] をクリックして、ターゲットグループのページを表示します。

ステップ 3 Firewall Threat Defense Virtual データインターフェイス IP を登録するターゲットグループを選択します。ターゲットグループの詳細ページが表示され、ターゲットフェールオーバー属性を有効にできます。

ステップ 4 [属性（Attributes）] メニューに移動します。

ステップ 5 [編集（Edit）] をクリックして属性を編集します。

ステップ 6 [フローの再調整（Rebalance flows）] スライダのボタンを右に切り替えてターゲットフェールオーバーを有効にし、ターゲットフェールオーバーや登録解除の際に既存のネットワークパケットを再調整して正常なターゲットノードに転送するように GWLB を設定します。

Azure でクラスタを展開する

Azure Gateway Load Balancer（GWLB）、または非ネイティブのロードバランサでクラスタを使用できます。Azure でクラスタを展開するには、Azure Resource Manager（ARM）テンプレートを使用して仮想マシンスケールセットを展開します。

GWLB ベースのクラスタ展開のサンプルトポロジ

図 *7: GWLB* を使用する着信トラフィックの導入例とトポロジ

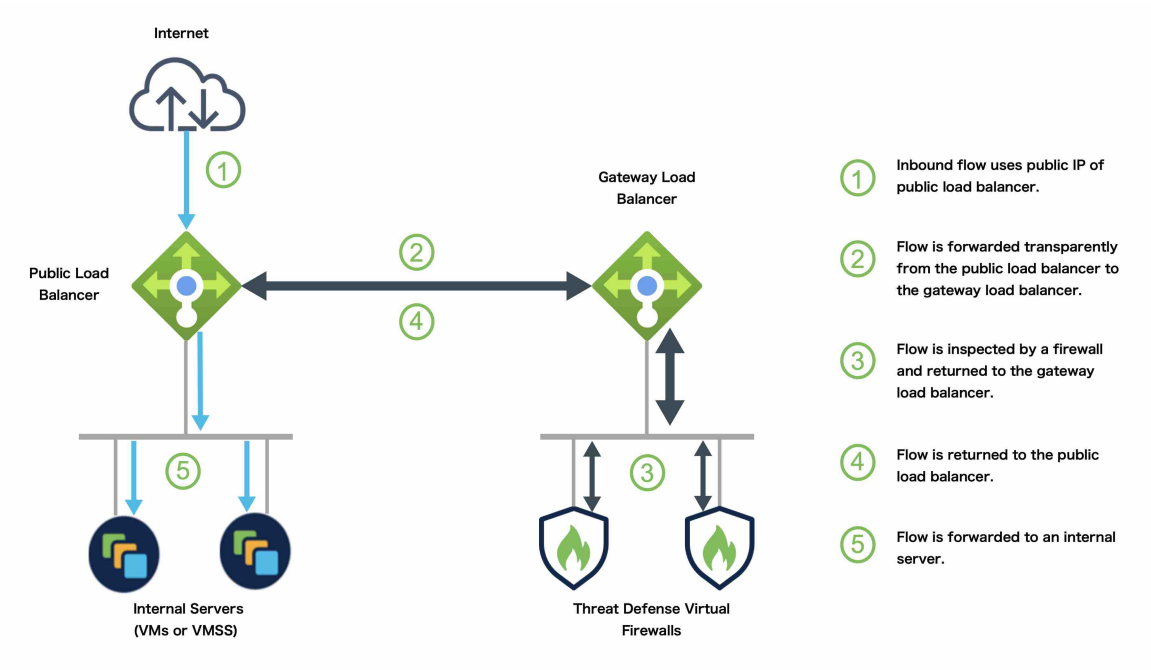
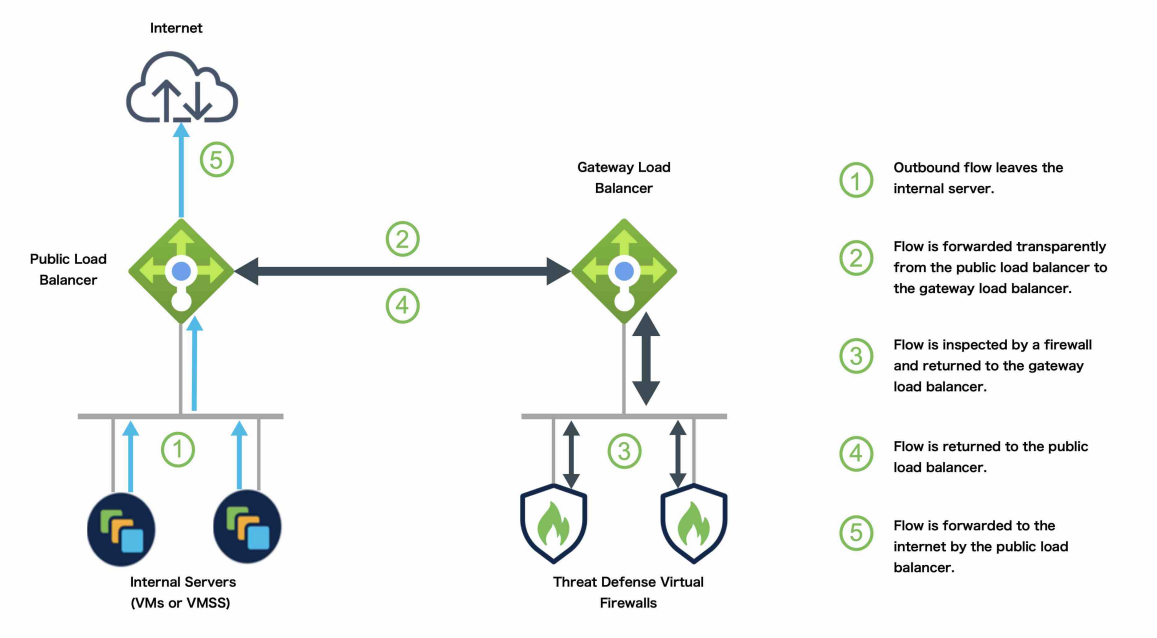


図 *8 : GWLB* を使用する発信トラフィックの導入例とトポロジ

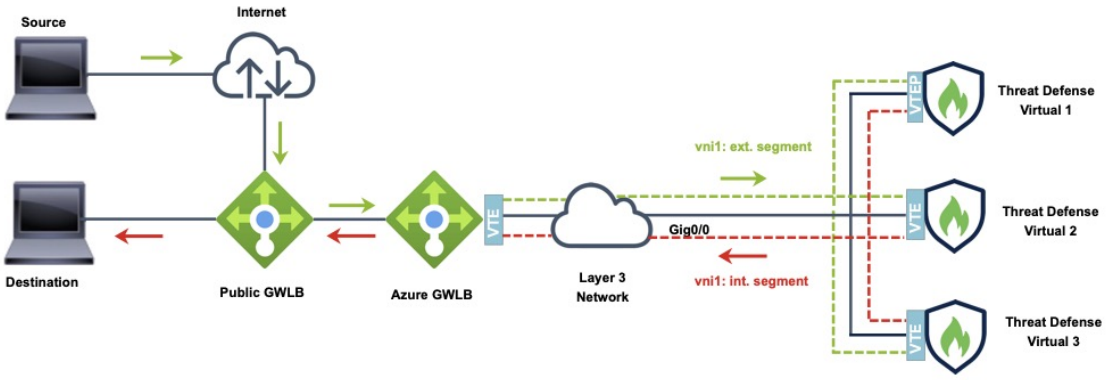


Azure ゲートウェイロードバランサおよびペアプロキシ

Azure サービスチェーンでは、Threat Defense Virtual がインターネットと顧客サービス間のパケットをインターセプトできる透過的なゲートウェイとして機能します。Threat Defense Virtual は、ペアプロキシの VXLAN セグメントを利用して、単一の NIC に外部インターフェイスと内部インターフェイスを定義します。

次の図は、外部 VXLAN セグメント上のパブリックゲートウェイロードバランサから Azure ゲートウェイロードバランサに転送されるトラフィックを示しています。ゲートウェイロードバランサは、複数の Threat Defense Virtual の間でトラフィックのバランスを取り、トラフィックをドロップするか、内部 VXLAN セグメント上のゲートウェイロードバランサに送り返す前に検査します。Azure ゲートウェイロードバランサは、トラフィックをパブリックゲートウェイロードバランサと宛先に送り返します。

図 *9*: ペアリングされたプロキシを使用した *Azure Gateway* ロードバランサ

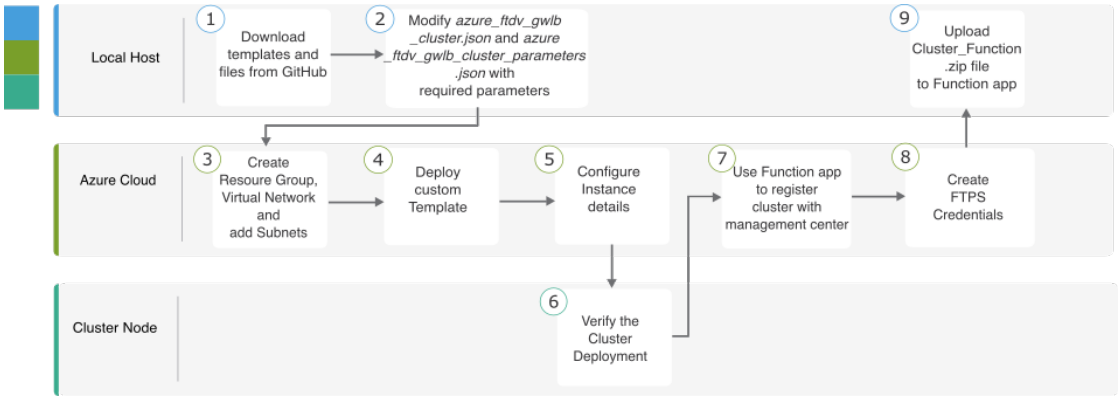


Traffic flow between GWLBe to GWLB (Geneve Single-Arm Proxy) in Azure

GWLB を使用して Azure で Threat Defense Virtual クラスタを展開するためのエンドツーエンドのプロセス

テンプレートベースの展開

次のフローチャートは、GWLB を使用した Azure での Threat Defense Virtual クラスタのテンプレートベース展開のワークフローを示しています。

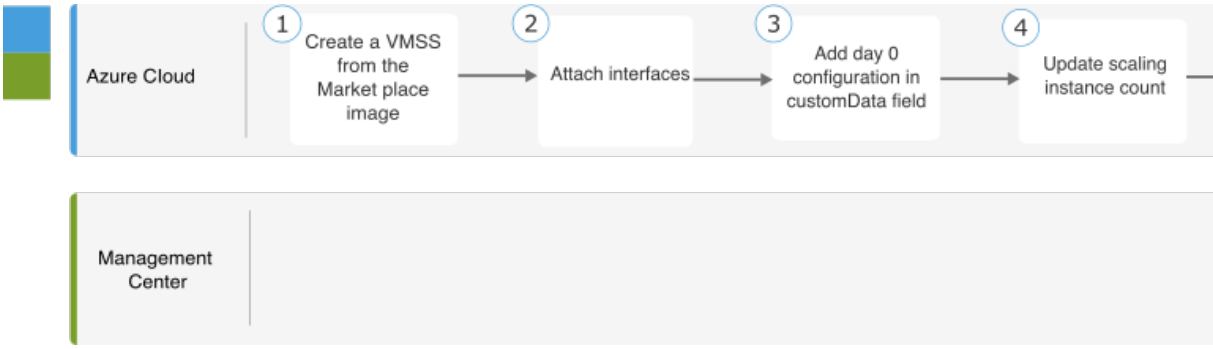


	ワークスペース	手順
1	ローカルホスト	GitHub からテンプレートとファイルをダウンロードします。
2	ローカルホスト	azure_ftdv_gwlb_cluster.json と azure_ftdv_gwlb_cluster_parameters.json を必要なパラメータで変更します。

	ワークスペース	手順
3	Azure Cloud	リソースグループ、仮想ネットワーク、およびサブネットを作成します。
4	Azure Cloud	カスタムテンプレートを展開します。
5	Azure Cloud	インスタンスの詳細を設定します。
6	クラスタノード	クラスタの展開を確認します。
7	Azure Cloud	Function アプリを使用して Management Center にクラスタを登録します。
8	Azure Cloud	FTPS のログイン情報を作成します。
9	ローカルホスト	Cluster_Function.zip ファイルを Function アプリにアップロードします。

手動展開

次のフローチャートは、GWLB を使用した Azure での Threat Defense Virtual クラスタの手動展開のワークフローを示しています。



	ワークスペース	手順
1	ローカルホスト	Marketplace イメージから VMSS を作成します。
2	ローカルホスト	インターフェイスを接続します。
3	ローカルホスト	[customData] フィールドに Day 0 構成を追加します。
4	ローカルホスト	スケーリングインスタンス数を更新します。
5	ローカルホスト	GWLB を設定します。

	ワークスペース	手順
6	Management Center	制御ノードを追加します。

テンプレート

以下のテンプレートは GitHub で入手できます。パラメータ値は、テンプレートで指定されたパラメータ名、および値であり、自明です。

Cisco Secure Firewall バージョン 7.4.1 以降では、診断インターフェイスを使用せずにクラスタを展開できます。外部、内部、管理、および CCL インターフェイスのみを使用してクラスタを展開するには、withoutDiagnostic テンプレート（azure_withoutDiagnostic_ftdv_gwlb_cluster_parameters.json および azure_withoutDiagnostic_ftdv_gwlb_cluster.json ファイル）を使用します。

診断インターフェイスで展開するテンプレート：

- azure_ftdv_gwlb_cluster_parameters.json：GWLB を使用する Firewall Threat Defense Virtual クラスタのパラメータを入力するためのテンプレート。
- azure_ftdv_gwlb_cluster.json：GWLB を使用する Firewall Threat Defense Virtual クラスタを展開するためのテンプレート。

診断インターフェイスなし で展開するテンプレート：

- azure_withoutDiagnostic_ftdv_gwlb_cluster_parameters.json：診断インターフェイスを使用せずに GWLB を展開する Firewall Threat Defense Virtual クラスタのパラメータを入力するテンプレート。
- azure_withoutDiagnostic_ftdv_gwlb_cluster.json：診断インターフェイスなしで GWLB を使用する Firewall Threat Defense Virtual クラスタを展開するためのテンプレート。

前提条件

- クラスタが Management Center に自動登録できるようにするには、Management Center でネットワーク管理者およびメンテナンスのユーザー権限を持つユーザーを作成します。これらの権限を持つユーザーは、REST API を使用できます。『Cisco Secure Firewall Management Center Administration Guide』を参照してください。
- テンプレートの展開時に指定するポリシー名と一致するアクセスポリシーを Management Center に追加します。
- Management Center Virtual が適切にライセンスされていることを確認します。
- クラスタが Management Center Virtual に追加されたら、次の手順を実行します。
  1. Management Center のプラットフォーム設定でヘルスチェックのポート番号を設定します。この設定の詳細については、「Platform Settings」を参照してください。

2. データトラフィックのスタティックルートを作成します。スタティックルートの作成の詳細については、「Add a Static Route」を参照してください。

   スタティックルートの設定例：

```
Network: any-ipv4
Interface: vxlan_tunnel
Leaked from Virtual Router: Global
Gateway: vxlan_tunnel_gw
Tunneled: false
Metric: 2
```

（注） *vxlan_tunnel_gw* は、データサブネットのゲートウェイ IP アドレスです。

Azure Resource Manager テンプレートを使用した Azure と GWLB でのクラスタの展開

カスタマイズされた Azure Resource Manager（ARM）テンプレートを使用して、Azure GWLB の仮想マシンスケールセットを展開します。

手順

ステップ 1 テンプレートを準備します。

a) GitHub リポジトリをローカルフォルダに複製します。https://github.com/CiscoDevNet/cisco-ftdv/tree/master/cluster/azure を参照してください。

b) azure_ftdv_gwlb_cluster.json と azure_ftdv_gwlb_cluster_parameters.json を必要なパラメータで変更します。

   または

   診断インターフェイスなしでクラスタを展開するために必要なパラメータを使用して、withoutDiagnostic テンプレート（*azure_withoutDiagnostic_ftdv_gwlb_cluster_parameters.json*、*azure_withoutDiagnostic_ftdv_gwlb_cluster.json*）を変更します。

ステップ 2 Azure ポータルにログイン：https://portal.azure.com。

ステップ 3 リソース グループを作成します。

a) [基本（Basics）] タブで、ドロップダウンリストから [サブスクリプション（Subscription）] および [リソースグループ（Resource group）] を選択します。

b) 必須の [リージョン（Region）] を選択します。

ステップ 4 管理、診断、外部、クラスタ制御リンク（CCL）の 4 つのサブネットを持つ仮想ネットワークを作成します。

Cisco Secure Firewall バージョン 7.4.1 以降では、診断インターフェイスを使用せずにクラスタを展開できます。外部、内部、管理、および CCL インターフェイスのみを使用してクラスタを展開するには、withoutDiagnostic テンプレート（azure_withoutDiagnostic_ftdv_gwlb_cluster_parameters.json および azure_withoutDiagnostic_ftdv_gwlb_cluster.json ファイル）を使用します。

a) 仮想ネットワークを作成します。

1. [基本（Basics）] タブで、ドロップダウンリストから [サブスクリプション（Subscription）] および [リソースグループ（Resource group）] を選択します。
2. 必須の [リージョン（Region）] を選択します。[次へ：IPアドレス（Next: IP addresses）] をクリックします。

[IPアドレス（IP Addresses）] タブで、[サブネットの追加（Add subnet）] をクリックし、管理、診断、データ、およびクラスタ制御リンクのサブネットを追加します。

Firewall Threat Defense Virtual 7.4.1 クラスタを診断インターフェイスなしで展開する場合は、診断サブネットの作成をスキップする必要があります。

b) サブネットを追加します。

ステップ 5 カスタムテンプレートを展開します。

a) **[作成（Create）] > [テンプレートの展開（Template deployment）]（カスタムテンプレートを使用して展開）**をクリックします。
b) [エディタで独自のテンプレートを構築する（Build your own template in the editor）] をクリックします。
c) [ファイルのロード（Load File）] をクリックし、**azure_ftdv_gwlb_cluster.json** または **azure_withoutDiagnostic_ftdv_gwlb_cluster.json**をアップロードします（診断インターフェイスなしでの展開を選択した場合）。
d) [保存（Save）] をクリックします。

ステップ 6 インスタンスの詳細を設定します。

a) 必要な値を入力し、[確認して作成（Review + create）] をクリックします。
b) 検証に合格したら、[作成（Create）] をクリックします。

ステップ 7 インスタンスの実行後、いずれかのノードにログインし、**show cluster info** コマンドを入力して、クラスタの展開を確認します。

図 10 : show cluster info

```
> show cluster info
Cluster gwlb-cluster-template-with-AN: On
    Interface mode: individual
Cluster Member Limit : 16
    This is "12" in state CONTROL_NODE
        ID          : 0
        Version     : 99.19(1)180
        Serial No.: 9AKGFV8VH4G
        CCL IP      : 10.1.1.12
        CCL MAC     : 000d.3a55.5470
        Module      : NGFWv
        Resource    : 8 cores / 28160 MB RAM
        Last join : 11:13:24 UTC Sep 5 2022
        Last leave: N/A
```

ステップ 8 Azure ポータルで、Function アプリをクリックしてクラスタを Firewall Management Center に登録します。

（注）
Function アプリを使用しない場合は、**[追加（Add）] > [デバイス（Device）]**（**[追加（Add）] > [クラスタ（Cluster）]** ではない）を使用して、制御ノードを Firewall Management Center に直接登録することもできます。その他のクラスタノードは自動的に登録されます。

ステップ 9 **[展開センター（Deployment Center）] > [FTPSのログイン情報（FTPS credentials）] > [ユーザースコープ（User scope）] > [ユーザー名とパスワードの設定（Configure Username and Password）]**をクリックして FTPS のログイン情報を作成し、[保存（Save）] をクリックします。

ステップ 10 ローカルの端末で次の **curl** コマンドを実行し、Cluster_Function.zip ファイルを Function アプリにアップロードします。

curl -X POST -u ユーザー名 **--data-binary @"Cluster_Function.zip" https://** *Function_App_Name***.scm.azurewebsites.net/api/zipdeploy**

（注）
curl コマンドは、実行が完了するまでに数分（2 分未満～ 3 分）かかる場合があります。

関数が Function アプリにアップロードされます。関数が開始され、ストレージアカウントのアウトキューにログが表示されます。Management Center へのデバイス登録が開始されます。

図 *11* : 機能

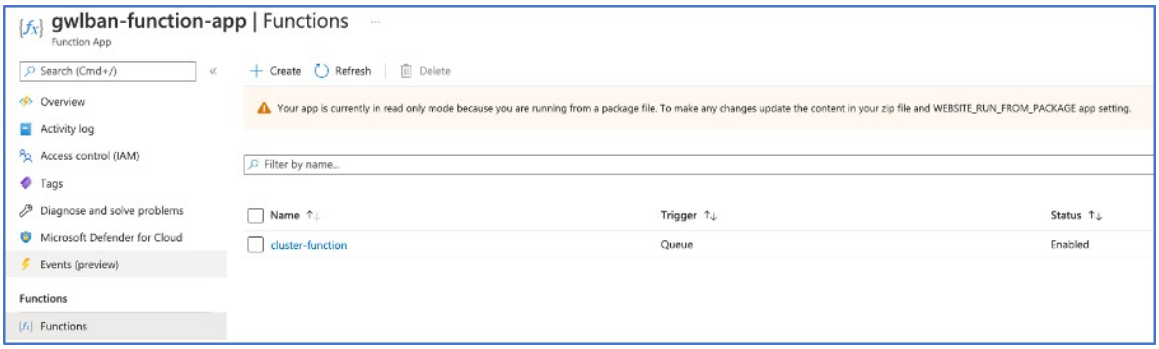


図 *12* : キュー

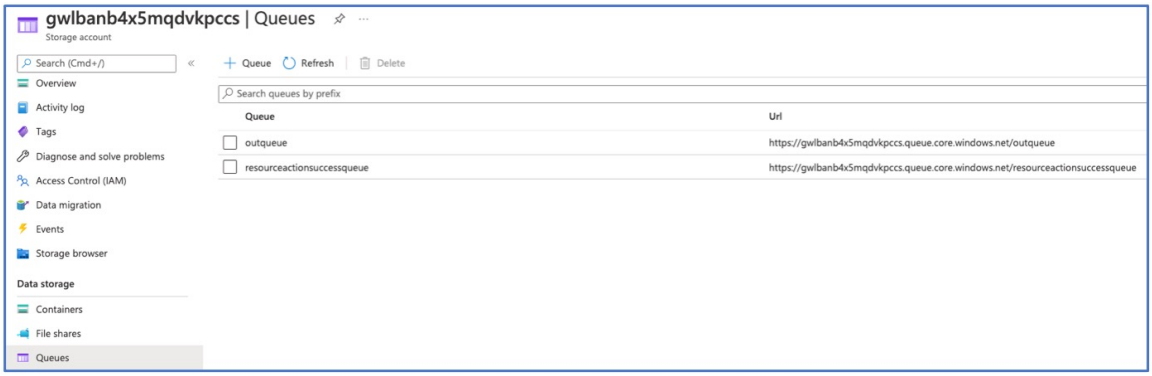
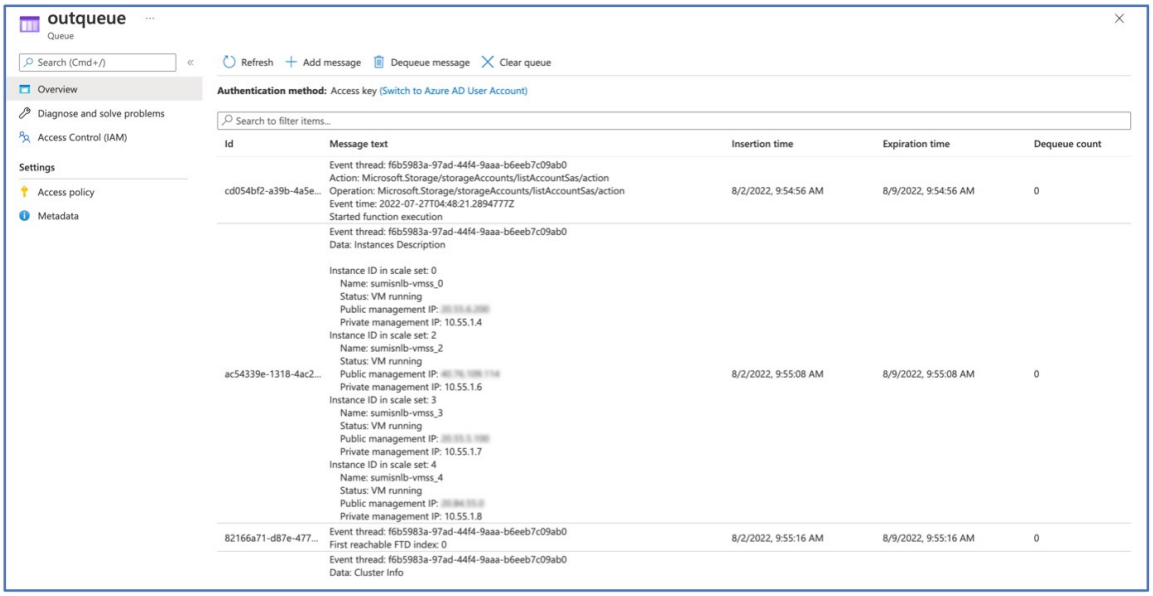
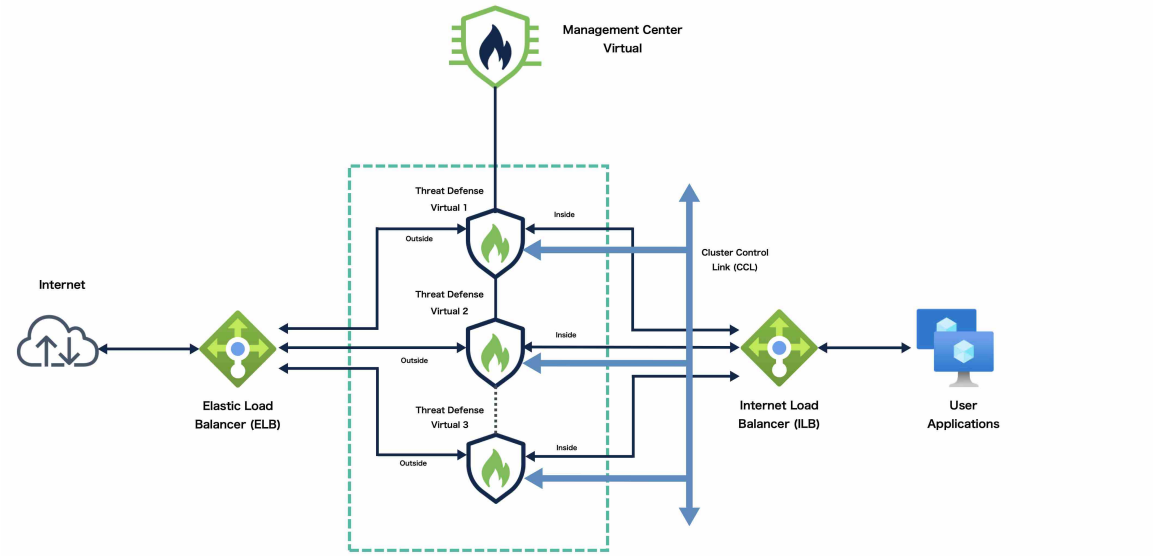


図 *13 :* アウトキュー



NLB ベースのクラスタ展開のサンプルトポロジ



このトポロジは、着信と発信の両方のトラフィックフローを示しています。Threat Defense Virtual クラスタは、内部ロードバランサと外部ロードバランサの間に挟まれています。Management Center Virtual インスタンスは、クラスタの管理に使用されます。

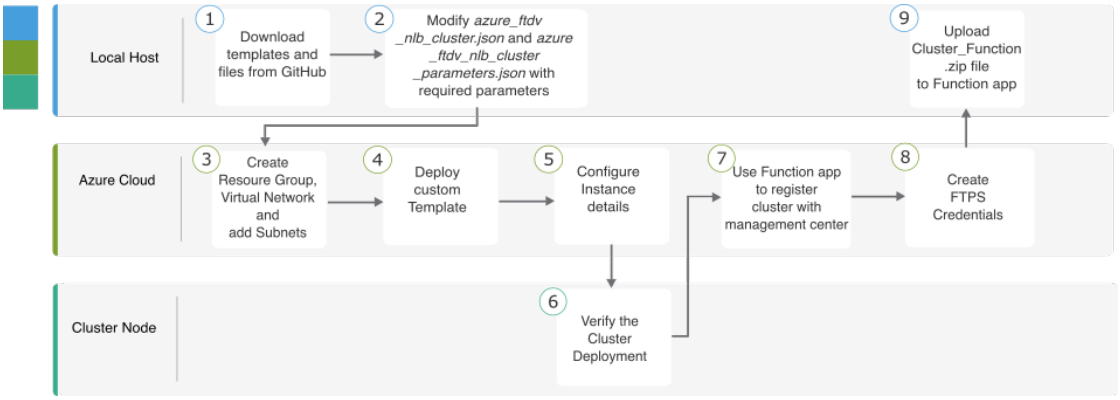
インターネットからの着信トラフィックは、外部ロードバランサに送られ、そこから Threat Defense Virtual クラスタにトラフィックが送信されます。トラフィックは、クラスタ内の Threat Defense Virtual インスタンスによって検査された後、アプリケーション VM に転送されます。

アプリケーション VM からの発信トラフィックは、内部ロードバランサに送信されます。その後、トラフィックは Threat Defense Virtual クラスタに転送され、インターネットに送信されます。

NLB を使用して Azure で Threat Defense Virtual クラスタを展開するためのエンドツーエンドのプロセス

テンプレートベースの展開

次のフローチャートは、NLB を使用した Azure での Threat Defense Virtual クラスタのテンプレートベース展開のワークフローを示しています。

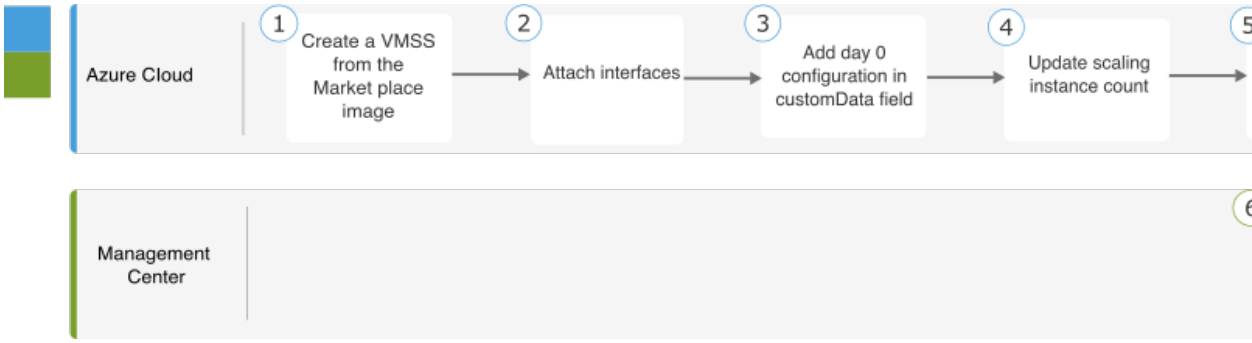


	ワークスペース	手順
1	ローカルホスト	GitHub からテンプレートとファイルをダウンロードします。
2	ローカルホスト	azure_ftdv_nlb_cluster.json と azure_ftdv_nlb_cluster_parameters.json を必要なパラメータで変更します。
3	Azure Cloud	リソースグループ、仮想ネットワーク、およびサブネットを作成します。
4	Azure Cloud	カスタムテンプレートを展開します。
5	Azure Cloud	インスタンスの詳細を設定します。
6	クラスタノード	クラスタの展開を確認します。

	ワークスペース	手順
7	Azure Cloud	Function アプリを使用して Management Center にクラスタを登録します。
8	Azure Cloud	FTPS のログイン情報を作成します。
9	ローカルホスト	Cluster_Function.zip ファイルを Function アプリにアップロードします。

手動展開

次のフローチャートは、NLB を使用した Azure での Threat Defense Virtual クラスタの手動展開のワークフローを示しています。



	ワークスペース	手順
1	ローカルホスト	Marketplace イメージから VMSS を作成します。
2	ローカルホスト	インターフェイスを接続します。
3	ローカルホスト	[customData] フィールドに Day 0 構成を追加します。
4	ローカルホスト	スケーリングインスタンス数を更新します。
5	ローカルホスト	NLB を設定します。
6	Management Center	制御ノードを追加します。

テンプレート

以下のテンプレートは GitHub で入手できます。パラメータ値は、テンプレートで指定されたパラメータ名、および値であり、自明です。

Cisco Secure Firewall バージョン 7.4.1 以降では、診断インターフェイスを使用せずにクラスタを展開できます。外部、内部、管理、および CCL インターフェイスのみを使用してクラスタを展開するには、withoutDiagnostic テンプレート（azure_withoutDiagnostic_ftdv_nlb_cluster_parameters.json および azure_withoutDiagnostic_ftdv_nlb_cluster.json ファイル）を使用します。

診断インターフェイスで展開するテンプレート：

- azure_ftdv_nlb_cluster_parameters.json：NLB を使用して Threat Defense Virtual クラスタのパラメータを入力するためのテンプレート。
- azure_ftdv_nlb_cluster.json：NLB を使用して Threat Defense Virtual クラスタを展開するためのテンプレート。

診断インターフェイスなし で展開するテンプレート：

- azure_withoutDiagnostic_ftdv_nlb_cluster_parameters.json：診断インターフェイスを使用しない NLB 展開で Firewall Threat Defense Virtual クラスタのパラメータを入力するためのテンプレート。
- azure_withoutDiagnostic_ftdv_nlb_cluster.json：診断インターフェイスなしの NLB を使用して Firewall Threat Defense Virtual クラスタを展開するためのテンプレート。

前提条件

- クラスタが Management Center に自動登録できるようにするには、Management Center でネットワーク管理者およびメンテナンスのユーザー権限を持つユーザーを作成します。これらの権限を持つユーザーは、REST API を使用できます。『Cisco Secure Firewall Management Center Administration Guide』を参照してください。
- テンプレートの展開時に指定するポリシー名と一致するアクセスポリシーを Management Center に追加します。
- Management Center Virtual が適切にライセンスされていることを確認します。
- クラスタが Management Center Virtual に追加されたら、次の手順を実行します。

1. Management Center のプラットフォーム設定でヘルスチェックのポート番号を設定します。この設定の詳細については、「Platform Settings」を参照してください。
2. 外部および内部インターフェイスからのトラフィックのスタティックルートを作成します。スタティックルートの作成の詳細については、「Add a Static Route」を参照してください。

   外部インターフェイスのスタティックルートの設定例：

```
Network: any-ipv4
Interface: outside
Leaked from Virtual Router: Global
Gateway: ftdv-cluster-outside
Tunneled: false
Metric: 10
```

（注） *ftdv-cluster-outside* は、外部サブネットのゲートウェイ IP アドレスです。

内部インターフェイスのスタティックルートの設定例：

```
Network: any-ipv4
Interface: inside
Leaked from Virtual Router: Global
Gateway: ftdv-cluster-inside-gw
Tunneled: false
Metric: 11
```

（注） *ftdv-cluster-inside-gw* は、内部サブネットのゲートウェイ IP アドレスです。

3. データトラフィックの NAT ルールを設定します。NAT ルールの設定の詳細については、「Network Address Translation」を参照してください。

Azure Resource Manager テンプレートを使用した Azure と NLB でのクラスタの展開

カスタマイズされた Azure Resource Manager（ARM）テンプレートを使用して、Azure NLB のクラスタを展開します。

手順

ステップ 1 テンプレートを準備します。

a) GitHub リポジトリをローカルフォルダに複製します。https://github.com/CiscoDevNet/cisco-ftdv/tree/master/cluster/azure を参照してください。

b) azure_ftdv_nlb_cluster.json と azure_ftdv_nlb_cluster_parameters.json を必要なパラメータで変更します。

診断インターフェイスなしでクラスタを展開するために必要なパラメータを使用して、withoutDiagnostic テンプレート *azure_withoutDiagnostic_ftdv_nlb_cluster_parameters.json* と *azure_withoutDiagnostic_ftdv_nlb_cluster.json* を変更します。

ステップ 2 Azure ポータルにログイン：https://portal.azure.com。

ステップ 3 リソース グループを作成します。

a) [基本（Basics）] タブで、ドロップダウンリストから [サブスクリプション（Subscription）] および [リソースグループ（Resource group）] を選択します。

b) 必須の [リージョン（Region）] を選択します。

ステップ 4 管理、診断、内部、外部、クラスタ制御リンクの 5 つのサブネットを持つ仮想ネットワークを作成します。

Cisco Secure Firewall バージョン 7.4.1 以降では、診断インターフェイスを使用せずにクラスタを展開できます。外部、内部、管理、およびクラスタ制御リンクのインターフェイスのみを使用してクラスタを展開するには、withinDiagnostic テンプレート（azure_withoutDiagnostic_ftdv_nlb_cluster_parameters.json および azure_withoutDiagnostic_ftdv_nlb_cluster.json ファイル）を使用します。

a) 仮想ネットワークを作成します。

1. [基本（Basics）] タブで、ドロップダウンリストから [サブスクリプション（Subscription）] および [リソースグループ（Resource group）] を選択します。
2. b）必須の [リージョン（Region）] を選択します。[次へ：IPアドレス（Next: IP addresses）] をクリックします。

b) サブネットを追加します。

[IPアドレス（IP Addresses）] タブで、[サブネットの追加（Add subnet）] をクリックし、管理、診断、内部、外部、およびクラスタ制御リンクのサブネットを追加します。

Firewall Threat Defense Virtual 7.4.1 クラスタを診断インターフェイスなしで展開する場合は、診断サブネットの作成をスキップする必要があります。

ステップ 5 カスタムテンプレートを展開します。

a) **[作成（Create）] > [テンプレートの展開（Template deployment）]（カスタムテンプレートを使用して展開）**をクリックします。
b) [エディタで独自のテンプレートを構築する（Build your own template in the editor）] をクリックします。
c) [ファイルのロード（Load File）] をクリックし、**azure_ftdv_nlb_cluster.json** または **azure_withoutDiagnostic_ftdv_nlb_cluster.json** をアップロードします（診断インターフェイスなしでの展開を選択した場合）。
d) [保存（Save）] をクリックします。

ステップ 6 インスタンスの詳細を設定します。

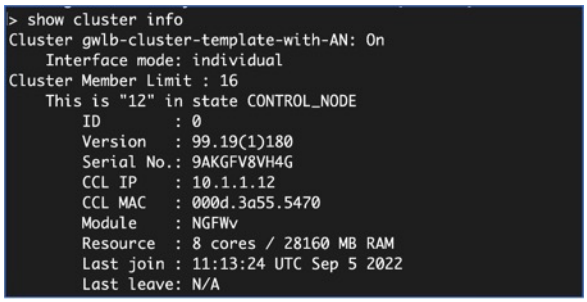
a) 必要な値を入力し、[確認して作成（Review + create）] をクリックします。

（注）
クラスタ制御リンクの開始アドレスと終了アドレスは、必要な数だけ指定してください（最大 16 個）。範囲を大きくすると、パフォーマンスに影響する可能性があります。

b) 検証に合格したら、[作成（Create）] をクリックします。

ステップ 7 インスタンスの実行後、いずれかのノードにログインし、**show cluster info** コマンドを使用して、クラスタの展開を確認します。

図 *14 : show cluster info*



ステップ 8 Azure ポータルで、Function アプリをクリックしてクラスタを Firewall Management Center に登録します。

（注）
Function アプリを使用しない場合は、[追加（Add）] > [デバイス（Device）]（[追加（Add）] > [クラスタ（Cluster）] ではない）を使用して、制御ノードを Management Center に直接登録することもできます。その他のクラスタノードは自動的に登録されます。

ステップ 9 **[展開センター（Deployment Center）]** > **[FTPSのログイン情報（FTPS credentials）]** > **[ユーザースコープ（User scope）]** > **[ユーザー名とパスワードの設定（Configure Username and Password）]**をクリックして FTPS のログイン情報を作成し、[保存（Save）] をクリックします。

ステップ 10 ローカルの端末で次の **curl** コマンドを実行し、Cluster_Function.zip ファイルを Function アプリにアップロードします。

curl -X POST -u ユーザー名 **--data-binary @"Cluster_Function.zip" https://** *Function_App_Name***.scm.azurewebsites.net/api/zipdeploy**

（注）
curl コマンドは、実行が完了するまでに数分（2 分未満～ 3 分）かかる場合があります。

関数が Function アプリにアップロードされます。関数が開始され、ストレージアカウントのアウトキューにログが表示されます。Management Center へのデバイス登録が開始されます。

Azure でのクラスタの手動展開

クラスタを手動で展開するには、Day0 構成を準備し、各ノードを展開してから制御ノードを Firewall Management Center に追加します。

Azure 向け Day 0 構成の作成

固定構成またはカスタマイズ構成のいずれかを使用できます。

Azure 向け固定構成を使用した Day 0 構成の作成

固定構成により、クラスタのブートストラップ構成が自動生成されます。

```
{
 "AdminPassword": "password",
 "FirewallMode": "Routed",
 "ManageLocally": "No",
 "Diagnostic": "OFF",    //For deployment of version 7.4.1 and later without Diagnostics
 template, set this parameter to OFF.
. "FmcIp": "<FMC_IP>",
 "FmcRegKey": "<REGISTRATION_KEY>",
 "FmcNatId": "<NAT_ID>",
 "Cluster": {
  "CclSubnetRange": "ip_address_start ip_address_end",
  "ClusterGroupName": "cluster_name",
  "HealthProbePort": "port_number",
  "GatewayLoadBalancerIP": "ip_address",
  "EncapsulationType": "vxlan",
  "InternalPort": "internal_port_number",
  "ExternalPort": "external_port_number",
  "InternalSegId": "internal_segment_id",
  "ExternalSegId": "external_segment_id"
 }
}
```

例

次に、Day 0 構成の例を示します。

```
{
 "AdminPassword": "password",
 "FirewallMode": "routed",
 "ManageLocally": "No",
 "Diagnostic": "OFF",    //For deployment of version 7.4.1 and later without Diagnostics
 template, set this parameter to OFF.
 "FmcIp":"<FMC_IP>",
 "FmcRegKey":"<REGISTRATION_KEY>",
 "FmcNatId":"<NAT_ID>",
 "Cluster": {
  "CclSubnetRange": "10.45.3.4 10.45.3.30",    //mandatory user input
  "ClusterGroupName": "ngfwv-cluster",         //mandatory user input
  "HealthProbePort": "7777",                   //mandatory user input
  "GatewayLoadBalancerIP": "10.45.2.4",         //mandatory user input
  "EncapsulationType": "vxlan",
  "InternalPort": "2000",
  "ExternalPort": "2001",
  "InternalSegId": "800",
  "ExternalSegId": "801"
 }
}
```

（注） 上記の設定をコピーして貼り付ける場合は、設定から **//mandatory user input** を必ず削除してください。

Azure ヘルスチェックの設定では、ここで設定した **HealthProbePort** を必ず指定してください。

CclSubnetRange 変数には、x.x.x.4 から始まる IP アドレスの範囲を指定します。クラスタリングに使用可能な IP アドレスが 16 個以上あることを確認します。開始 IP アドレスと終了 IP アドレスの例を次に示します。

表 *3:* 開始 *IP* アドレスと終了 *IP* アドレスの例

CIDR	開始 IP アドレス	終了 IP アドレス
10.1.1.0/27	10.1.1.4	10.1.1.30
10.1.1.32/27	10.1.1.36	10.1.1.62
10.1.1.64/27	10.1.1.68	10.1.1.94
10.1.1.96/27	10.1.1.100	10.1.1.126
10.1.1.128/27	10.1.1.132	10.1.1.158
10.1.1.160/27	10.1.1.164	10.1.1.190
10.1.1.192/27	10.1.1.196	10.1.1.222
10.1.1.224/27	10.1.1.228	10.1.1.254

Azure 向けカスタマイズ構成を使用した Day 0 構成の作成

コマンドを使用して、クラスタのブートストラップ設定をすべて入力できます。

```
{
 "AdminPassword": "password",
 "FirewallMode": "Routed",
 "ManageLocally": "No",
 "Diagnostic": "OFF",    //For deployment of version 7.4.1 and later without Diagnostics
 template, set this parameter to OFF.
 "FmcIp": "<FMC_IP>",
 "FmcRegKey": "<REGISTRATION_KEY>",
 "FmcNatId": "<NAT_ID>",
 "Cluster": {
  "CclSubnetRange": "ip_address_start ip_address_end",
  "ClusterGroupName": "cluster_name",
  "HealthProbePort": "port_number",
  "GatewayLoadBalancerIP": "ip_address",
  "EncapsulationType": "vxlan",
  "InternalPort": "internal_port_number",
  "ExternalPort": "external_port_number",
  "InternalSegId": "internal_segment_id",
  "ExternalSegId": "external_segment_id"
 }
}
```

例

以下に、**バージョン 7.4 以降**の Day 0 構成の例を示します。

```
{
 "AdminPassword": "Sup3rnatural",
 "Hostname": "clusterftdv",
 "FirewallMode": "routed",
 "ManageLocally": "No",
```

```
 "Diagnostic": "OFF",    //For deployment of version 7.4.1 and later without Diagnostics
 template, set this parameter to OFF.
 "FmcIp": "<FMC_IP>",
 "FmcRegKey": "<REGISTRATION_KEY>",
 "FmcNatId": "<NAT_ID>",
 "run_config": [
  "cluster interface-mode individual force",
  "policy-map global_policy",
  "class inspection_default",
  "no inspect h323 h225",
  "no inspect h323 ras",
  "no inspect rtsp",
  "no inspect skinny",
  "interface Management0/0",
  "management-only",
  "nameif management",
  "security-level 0",
  "ip address dhcp",
  "interface GigabitEthernet0/0",
  "no shutdown",
  "nameif vxlan_tunnel",
  "security-level 0",
  "ip address dhcp",
  "interface GigabitEthernet0/1",
  "no shutdown",
  "nve-only cluster",
  "nameif ccl_link",
  "security-level 0",
  "ip address dhcp",
  "interface vni1",
  "description Clustering Interface",
  "segment-id 1",
  "vtep-nve 1",
  "interface vni2",
  "proxy paired",
  "nameif GWLB-backend-pool",
  "internal-segment-id 800",
  "external-segment-id 801",
  "internal-port 2000",
  "external-port 2001",
  "security-level 0",
  "vtep-nve 2",
  "object network ccl#link",
  "range 10.45.3.4 10.45.3.30",                          //mandatory user input
  "object-group network cluster#group",
  "network-object object ccl#link",
  "nve 1 ",
  "encapsulation vxlan",
  "source-interface ccl_link",
  "peer-group cluster#group",
  "nve 2 ",
  "encapsulation vxlan",
  "source-interface vxlan_tunnel",
  "peer ip <GatewayLoadbalancerIP>",
  "cluster group ftdv-cluster",                          //mandatory user input
  "local-unit 1",
  "cluster-interface vni1 ip 1.1.1.1 255.255.255.0",
  "priority 1",
  "enable",
  "mtu vxlan_tunnel 1454",
  "mtu ccl_link 1454"
 ]
}
```

以下に、**バージョン 7.3 以前**の Day 0 構成の例を示します。

```
{
 "AdminPassword": "Sup3rnatural",
 "Hostname": "clusterftdv",
 "FirewallMode": "routed",
 "ManageLocally": "No",
 "FmcIp": "<FMC_IP>",
 "FmcRegKey": "<REGISTRATION_KEY>",
 "FmcNatId": "<NAT_ID>",
 "run_config": [
  "cluster interface-mode individual force",
  "policy-map global_policy",
  "class inspection_default",
  "no inspect h323 h225",
  "no inspect h323 ras",
  "no inspect rtsp",
  "no inspect skinny",
  "interface Management0/0",
  "management-only",
  "nameif management",
  "security-level 0",
  "ip address dhcp",
  "interface GigabitEthernet0/0",
  "no shutdown",
  "nameif vxlan_tunnel",
  "security-level 0",
  "ip address dhcp",
  "interface GigabitEthernet0/1",
  "no shutdown",
  "nve-only cluster",
  "nameif ccl_link",
  "security-level 0",
  "ip address dhcp",
  "interface vni1",
  "description Clustering Interface",
  "segment-id 1",
  "vtep-nve 1",
  "interface vni2",
  "proxy paired",
  "nameif GWLB-backend-pool",
  "internal-segment-id 800",
  "external-segment-id 801",
  "internal-port 2000",
  "external-port 2001",
  "security-level 0",
  "vtep-nve 2",
  "object network ccl#link",
  "range 10.45.3.4 10.45.3.30",                          //mandatory user input
  "object-group network cluster#group",
  "network-object object ccl#link",
  "nve 1 ",
  "encapsulation vxlan",
  "source-interface ccl_link",
  "peer-group cluster#group",
  "nve 2 ",
  "encapsulation vxlan",
  "source-interface vxlan_tunnel",
  "peer ip <GatewayLoadbalancerIP>",
  "cluster group ftdv-cluster",                          //mandatory user input
  "local-unit 1",
  "cluster-interface vni1 ip 1.1.1.1 255.255.255.0",
  "priority 1",
```

```
  "enable",
  "mtu vxlan_tunnel 1454",
  "mtu ccl_link 1554"
 ]
}
```

（注） 上記の設定をコピーして貼り付ける場合は、設定から **//mandatory user input** を必ず削除してください。

クラスタノードの手動展開：GWLB ベースの展開

クラスタが形成されるようにクラスタノードを展開します。

手順

ステップ 1 **az vmss create** CLI を使用して、インスタンス数が 0 の Marketplace イメージから仮想マシンスケールセットを作成します。

az vmss create --resource-group <*ResourceGroupName*> **--name** <*VMSSName*> **--vm-sku** <*InstanceSize*> **--image** <*FTDvImage*> **--instance-count 0 --admin-username** <*AdminUserName*> **--admin-password** <*AdminPassword*> **--plan-name** <*ftdv-azure-byol/ftdv-azure-payg*> **--plan-publisher cisco --plan-product cisco-ftdv --plan-promotion-code** <*ftdv-azure-byol/ftdv-azure-payg*> **--vnet-name** <*VirtualNetworkName*> **--subnet** <*MgmtSubnetName*>

ステップ 2 3 つのインターフェイス（診断、データ、およびクラスタ制御リンク）を接続します。

ステップ 3 作成した仮想マシンスケールセットに移動し、次の手順を実行します。

a) [オペレーティングシステム（Operating system）] セクションで、[customData] フィールドに Day 0 構成を追加します。

b) [保存（Save）] をクリックします。

c) [スケーリング（Scaling）] セクションで、インスタンス数を必要なクラスタノードで更新します。インスタンス数は、最小 1、最大 16 の範囲に設定できます。

ステップ 4 Azure ゲートウェイロードバランサを設定します。詳細については、「Azure ゲートウェイロードバランサを使用した Auto Scale の導入例」を参照してください。

ステップ 5 Firewall Management Center に制御ノードを追加します。「Management Center へのクラスタの追加（手動展開） （96 ページ）」を参照してください。

クラスタノードの手動展開：NLB ベースの展開

クラスタが形成されるようにクラスタノードを展開します。

手順

ステップ 1 **az vmss create** CLI を使用して、インスタンス数が 0 の Marketplace イメージから仮想マシンスケールセットを作成します。

az vmss create --resource-group <*ResourceGroupName*> **--name** <*VMSSName*> **--vm-sku** <*InstanceSize*> **--image** <*FTDvImage*> **--instance-count 0 --admin-username** <*AdminUserName*> **--admin-password** <*AdminPassword*> **--plan-name** <*ftdv-azure-byol/ftdv-azure-payg*> **--plan-publisher cisco --plan-product cisco-ftdv --plan-promotion-code** <*ftdv-azure-byol/ftdv-azure-payg*> **--vnet-name** <*VirtualNetworkName*> **--subnet** <*MgmtSubnetName*>

ステップ 2 4 つのインターフェイス（診断、内部、外部、およびクラスタ制御リンク）を接続します。

ステップ 3 作成した仮想マシンスケールセットに移動し、次の手順を実行します。

a) [オペレーティングシステム（Operating system）] セクションで、[customData] フィールドに Day 0 構成を追加します。

b) [保存（Save）] をクリックします。

c) [スケーリング（Scaling）] セクションで、インスタンス数を必要なクラスタノードで更新します。インスタンス数は、最小 1、最大 16 の範囲に設定できます。

ステップ 4 Management Center に制御ノードを追加します。「Management Center へのクラスタの追加（手動展開） （96 ページ）」を参照してください。

Azure でのトラブルシューティング クラスタ展開

- 問題：トラフィックフローがない

  トラブルシューティング：

  - GWLB で展開された Threat Defense Virtual インスタンスの正常性プローブステータスが正常かどうかを確認します。
  - Threat Defense Virtual インスタンスの正常性プローブステータスが異常である場合：
    - Management Center Virtual でスタティックルートが設定されているかどうかを確認します。
    - デフォルトゲートウェイがデータサブネットのゲートウェイ IP であるかどうかを確認します。
    - Threat Defense Virtual インスタンスが正常性プローブトラフィックを受信しているかどうかを確認します。
    - Management Center Virtual で設定されたアクセスリストが正常性プローブトラフィックを許可しているかどうかを確認します。

- 問題：クラスタが形成されていない

トラブルシューティング：

- nve-only クラスタインターフェイスの IP アドレスを確認します。他のノードの nve-only のクラスタインターフェイスにピン可能であることを確認します。
- nve-only のクラスタインターフェイスの IP アドレスが、オブジェクトグループの一部であることを確認します。
- NVE インターフェイスがオブジェクトグループで設定されていることを確認します。
- クラスタグループのクラスタインターフェイスに適切な VNI インターフェイスがあることを確認します。この VNI インターフェイスには、対応するオブジェクトグループを持つ NVE があります。
- ノードが相互にピン可能であることを確認します。各ノードに独自のクラスタインターフェイス IP があるため、これらは相互にピン可能である必要があります。
- テンプレート展開中に指定された CCL サブネットの開始アドレスと終了アドレスが正しいかどうかを確認します。開始アドレスは、サブネット内で使用可能な最初の IP アドレスで始まる必要があります。たとえばサブネットが 192.168.1.0/24 の場合、開始アドレスは 192.168.1.4 である必要があります（最初の 3 つの IP アドレスは Azure によって予約されています）。
- Management Center Virtual に有効なライセンスがあるかどうかを確認します。

- 問題：同じリソースグループに再度リソースを展開しているときにロールに関連するエラーが発生する。

  トラブルシューティング：端末で次のコマンドを使用して、以下のロールを削除します。

  エラー メッセージ：

```
"error": {
"code": "RoleAssignmentUpdateNotPermitted",
"message": "Tenant ID, application ID, principal ID, and scope are not allowed to
be
updated.”}
```

  - **az role assignment delete --resource-group** <リソースグループ名> **--role "Storage Queue Data Contributor"**
  - **az role assignment delete --resource-group** <リソースグループ名> **--role "Contributor"**

Firewall Threat Defense Virtual Azure でのクラスタリングのオートスケール ソリューション

Azure リージョンでの一般的なクラスタ展開には、定義された数の Firewall Threat Defense Virtual インスタンス（ノード）が含まれます。Azure リージョンのトラフィックが変化しても、ノードのダイナミック スケーリング（オートスケール）が行われず、以前からのクラスタ配置のま

まだと、リソースが十分に活用されなかったり、遅延を引き起こしたりします。シスコは、Azure リージョンのノードのダイナミック スケーリングをサポートする Firewall Threat Defense Virtual クラスタリング向けにオートスケール ソリューションをバージョン 7.7 以降で提供しています。このソリューションにより、ネットワーク トラフィックに基づいてクラスタからノードを追加または削除して、スケールインまたはスケールアウトを行えます。CPU やメモリのメトリックなどの Azure VMSS メトリックからのリソース使用率の統計に基づくロジックを使用して、クラスタに対してノードを動的に追加または削除します。

Azure の Auto Scale ソリューションを使用した Firewall Threat Defense Virtual クラスタリングは、ネットワーク ロードバランサ（NLB またはサンドイッチ トポロジ）とゲートウェイ ロードバランサ（GWLB）の両方をサポートします。トポロジの例 （58 ページ）を参照してください。

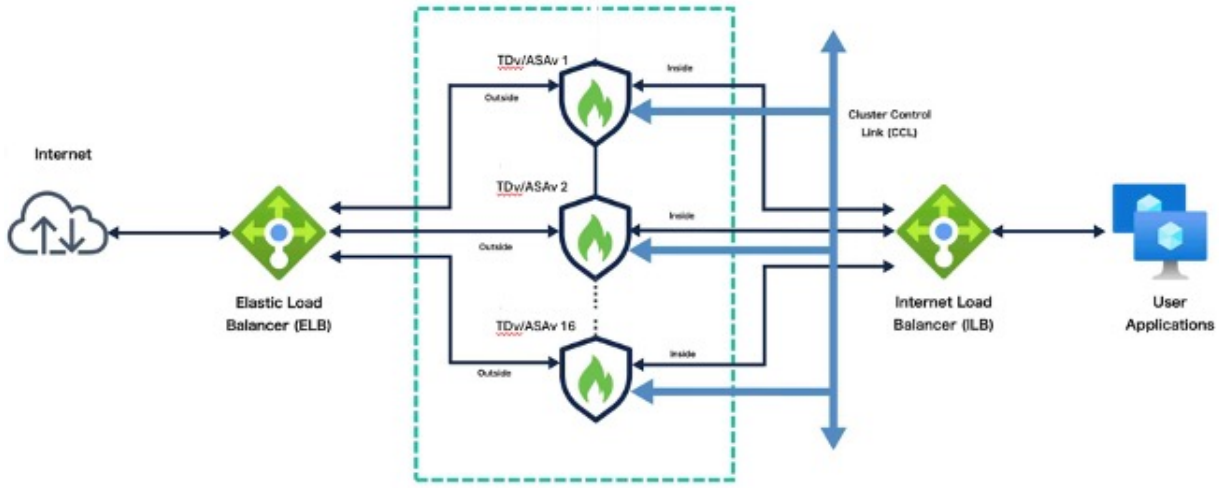
Cisco では、NLB や GWLB を使用して Firewall Threat Defense Virtual クラスタを Azure にオートスケールして展開するための個別の Azure Resource Manager（ARM）テンプレートと、関数アプリや論理アプリなど、Azure サービスを展開するためのインフラストラクチャ テンプレートと構成テンプレートを提供しています。

トポロジの例

Firewall Threat Defense Virtual サンドイッチ トポロジ（ネットワーク ロードバランサ）を使用した Azure でのオートスケールのクラスタリング

サンドイッチ トポロジ（NLB）を使用する Azure のオートスケール対応 Firewall Threat Defense Virtual クラスタリングのユースケースは、Azure の内部ロードバランサ（ILB）と Azure の外部ロードバランサ（ELB）にサンドイッチされるように Firewall Threat Defense Virtual スケールセットを配置する、自動水平スケーリング ソリューションです。

このトポロジでは、Firewall Threat Defense Virtual は、管理、内部、外部、および CCL サブネットの *4* つの インターフェイスのみを使用します。



Firewall Threat Defense Virtual サンドイッチトポロジ（NLB）を使用した Azure でのオートスケール クラスタリング

以下では、NLB 機能を使用して Azure で のオートスケーリングを行う Firewall Threat Defense Virtual クラスタのフローの概要を示します。

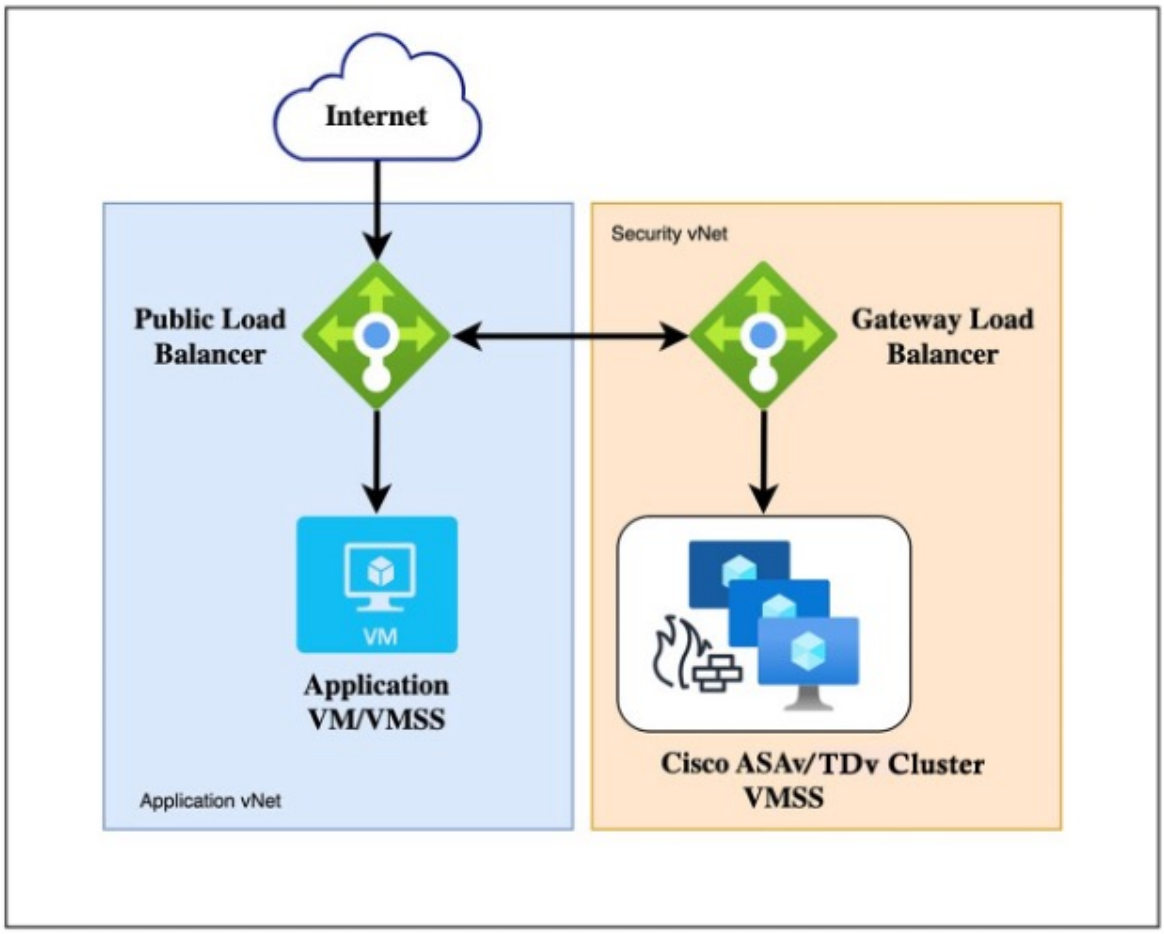
- ELB は、インターネットからのトラフィックをスケールセット内の Firewall Threat Defense Virtual インスタンスに分散させます。その後、ファイアウォールがアプリケーションにトラフィックを転送します。
- ILB は、アプリケーションからのアウトバウンド インターネット トラフィックをスケールセット内の Firewall Threat Defense Virtual インスタンスに分散させます。その後、ファイアウォールがインターネットにトラフィックを転送します。
- ネットワークパケットが、単一の接続で両方（内部および外部）のロードバランサを通過することはありません。
- スケールセット内の Firewall Threat Defense Virtual インスタンスの数は、負荷条件に基づいて自動的にスケーリングおよび設定されます。

Firewall Threat Defense Virtual ゲートウェイロードバランサを使用した Azure でのオートスケール クラスタリング

オートスケールソリューションを使用した Azure ゲートウェイロードバランサ（GWLB）と Firewall Threat Defense Virtual クラスタの統合により、クラスタセットアップでのインスタンスの展開、管理、およびスケーリングが簡素化されます。Azure ゲートウェイ ロードバランサ（GWLB）は、アプリケーションサーバーなどの Azure VM との間のインターネットトラフィックが、ルーティングの変更を必要とせずに Secure Firewall によって検査されるようにします。また、この統合により、運用の複雑さが軽減され、ファイアウォールでのトラフィックの単一のエントリポイントとエグジットポイントが提供されます。アプリケーションとインフラストラクチャは、送信元 IP アドレスの可視性を維持できます。一部の環境では、この可視性が非常に重要です。

Firewall Threat Defense Virtual は、この使用例では、管理、データ、CCL インターフェイスの *3* つの インターフェイスのみを使用します。

（注）
- Azure GWLB を展開する場合、ネットワークアドレス変換（NAT）は必要ありません。
- IPv4 だけがサポートされます。



以下では、GWLB 機能を使用して、Azure でオートスケーリングを行う Firewall Threat Defense Virtualクラスタのフローの概要を説明します。

- インターネットからの着信トラフィックは、GWLB エンドポイントに送られ、そこから GWLB にトラフィックが送信されます。
- その後、トラフィックは Firewall Threat Defense Virtual クラスタにルーティングされます。
- トラフィックは、クラスタ内の Firewall Threat Defense Virtual インスタンスによって検査された後、アプリケーション VM に転送されます。

Prerequisites

- Ensure that you have Owner role in the Azure subscription.
- Create the Azure Resource Group. Ensure that the Azure Virtual Network along with the necessary subnets are created.
  - Interfaces for NLB-based cluster : Management, Diagnostic, Inside, Outside, CCL and the function app.

- Interfaces for GWLB-based cluster : Management, Diagnostic, Data, CCL and the function app.

- On the Management Center:
  - Ensure that Management Center Virtual is licensed correctly.
  - Create the access control policy.
  - Create the Security Zone (SZ) object for the interfaces. For NLB based cluster, create the SZ for inside and outside interfaces. For GWLB-based cluster, create the SZ for the data interface.
  - Create a separate user name and password for the azure function to add the Threat Defense Virtual instances to the Management Center Virtual and configure the instances.
- Install the Azure CLI on your local system.
- Download the Azure Clustering Autoscale repository from GitHub to your local computer and run the command **python3 make.py build** to create the Azure functions zip file.

Azure での Firewall Threat Defense Virtual クラスタリングのオートスケール ロジック

スケーリングポリシー

オートスケールを備えたクラスタでは、ノードのスケーリングは次のポリシーに基づいて決定されます。

- スケーリング ポリシー 1：1 つのクラスタ ノードがリソース使用率の制限を超える場合。
- スケーリング ポリシー 2：すべてのノードの全体的な平均リソース使用率による。

スケールアウト

スケールアウトとは、トラフィック負荷のしきい値がクラスタのいずれかのノードに設定されている CPU またはメモリの制限を超えたときに、クラスタに新しいノードを追加するプロセスです。

次に、スケールアウト中にクラスタに新しいノードを追加するプロセスを示します。

1. 新しい Firewall Threat Defense Virtual インスタンスが起動します。
2. Firewall Threat Defense Virtual に適切な設定が適用されます。
3. 適切なライセンスが適用されます。
4. 新しい Firewall Threat Defense Virtual インスタンスがクラスタに追加されます。

スケールアウトプロセス中に新しい Firewall Threat Defense Virtual インスタンスの設定が失敗した場合（確率は低い）、失敗したインスタンスは終了し、新しいインスタンスが起動して設定されます。

スケールイン

スケールインは、設定されたスケールインしきいに達した場合、およびクラスタインスタンスの合計数が最小クラスタ サイズを超えた場合に、クラスタからノードを削除するプロセスです。

次に、スケールイン中にクラスタ内のノードを終了するプロセスを示します。

1. CPU またはメモリ使用率が最も低い Firewall Threat Defense Virtual インスタンスを、VMSS メトリックを使用して識別します。
2. 使用率が同じ最小のインスタンスが複数ある場合、VMSS の VM インデックスが高いインスタンスがスケールイン用に選択されます。
3. このインスタンスへの新しい接続は、適切な設定とポリシーによって無効になります。
4. インスタンスがスマート ライセンスから登録解除されます（BYOL に該当）。
5. インスタンスが終了します。

Azure 関数（Function App）

Function アプリケーションは、Firewall Threat Defense Virtual クラスタを有効化し、Management Center にクラスタを登録するのに役立ちます。Function アプリケーションは、自動スケール展開を使用した Firewall Threat Defense Virtual クラスタリングのホスティング プランを選択するのにも役立ちます。

次の 2 種類のホスティング プランが提供されています。

- **消費**
  - これは、オートスケールを使用した Firewall Threat Defense Virtual クラスタリングのデフォルトのホスティング プランです。
  - このプランでは、 リージョンの Azure データセンター IP アドレスへの SSH ポートを開くことにより、Function アプリが Firewall Threat Defense Virtual インスタンスに接続できます。
- **プレミアム**
  - 展開時に Function アプリに対してこのホスティング プランを選択できます。
  - このプランでは、Function アプリにネットワークアドレス変換（NAT）ゲートウェイを追加して、Function アプリのアウトバウンド IP アドレスを制御できます。このプランでは、NAT ゲートウェイの固定 IP アドレスからのみ Firewall Threat Defense Virtual インスタンスへの SSH アクセスを許可するため、セキュリティが強化されます。

Auto Scale ソリューションのコンポーネントの概要については、*Cisco Secure Firewall Threat Defense Virtual* スタートアップガイドの Auto Scale ソリューションのコンポーネントを参照してください。

GitHub での展開とインフラストラクチャのテンプレート

シスコでは、Function App、Logic App、自動スケーリンググループなどの複数の Azure サービスを使用して Firewall Threat Defense Virtual クラスタの Auto Scaling グループを展開するための Azure Resource Manager（ARM）テンプレートとスクリプトを提供しています。

Firewall Threat Defense Virtual クラスタのオートスケール ソリューションは、以下の内容を提供する ARM テンプレートベースの導入です。

- 関数アプリを使用した管理センターによる Firewall Threat Defense Virtual インスタンスの登録と登録解除の完全な自動化。
- スケールアウトされた Threat Defense Virtual インスタンスへの NAT ポリシー、アクセス コントロール ポリシー、およびルートの自動適用。
- GWLB および NLB ロードバランサのサポート。
- 管理センターを使用する必要があります。デバイスマネージャはサポートされていません。

Firewall Threat Defense Virtual オートスケールソリューションテンプレートを使用したクラスタリング

Azure リソース マネージャ（ARM）テンプレート

クラスタに Azure で使用している（NLB または GWLB）ロードバランサに基づいて、オートスケール ソリューション用に 2 セットのテンプレートが用意されています。

GitHub では、次のテンプレートを使用できます。

- NLB： `azure_ftdv_nlb_cluster.json.json` を使用した Firewall Threat Defense Virtual クラスタリング用のオートスケール ソリューション テンプレートは、`arm-templates` フォルダにあります。
- GWLB： `azure_ftdv_gwlb_cluster.json` を使用した Firewall Threat Defense Virtual クラスタリング用のオートスケール ソリューション テンプレートは、`arm-templates` フォルダにあります。

Azure インフラストラクチャと構成の設定

- Firewall Threat Defense Virtual インスタンスでクラスタを有効にする機能アプリ： `cluster_functions.zip`。
- Firewall Threat Defense Virtual の展開、スケールイン、およびスケールアウト ワークフロー用の論理アプリ コード： `logic_app.txt`。

入力パラメータ

次の表に、テンプレートパラメータおよび例を示します。各パラメータの値を決めたら、Azure サブスクリプションに Azure Resource Manager（ARM） テンプレートを展開するときに、各パラメータを使用して Firewall Threat Defense Virtual を作成できます。Azure 向けの GWLB を使用したオートスケールソリューションによるクラスタリングでは、テンプレートで追加の入力パラメータを設定する必要があるため、ネットワーキングインフラストラクチャも作成されます。パラメータの意味は一目瞭然なので説明を省略します。

表 *4:* テンプレートパラメータ

パラメータ名	使用できる値/タイプ	説明	リソースの作成タイプ
resourceNamePrefix	文字列*（3 ～ 10 文字）	すべてのリソースは、このプレフィックスを含む名前で作成されます。 注：小文字のみを使用してください。 例：ftdv	新規作成
virtualNetworkRg	文字列	仮想ネットワークのリソースグループの名前。 例：cisco-virtualnet-rg	既存
virtualNetworkName	文字列	仮想ネットワーク名（作成済み） 例：cisco-virtualnet	既存
virtualNetworkCidr	CIDR 形式 x.x.x.x/y	仮想ネットワークの CIDR（作成済み）	既存
mgmtSubnet	文字列	管理サブネット名（作成済み） 例：cisco-mgmt-subnet	既存
dataSubnet	文字列	データ サブネット名（作成済み） 例：cisco-data-subnet	
cclSubnet	文字列	クラスタ制御リンクのサブネット名。 例：cisco-ccl-subnet	

パラメータ名	使用できる値/タイプ	説明	リソースの作成タイプ
cclSubnetStartAddr	文字列	CCL サブネット IP アドレスの範囲開始。 例：3.4.5.6	
cclSubnetEndAddr	文字列	CCL サブネット IP アドレスの範囲終了。 例：5.6.7.8	
gwlbIP	文字列	GWLB は既存のデータサブネットに作成されます。 例：10.0.2.4	
dataNetworkGatewayIp	文字列	データ サブネットのゲートウェイ IP アドレス。 例：10.0.2.7	
outsideSecurityZoneName	文字列	管理センターで作成されたセキュリティ ゾーン オブジェクト名 例：outside-sz	
TDvmManagementUserName	文字列	TDv 管理の管理者ユーザー名。 ユーザー名として「admin」を指定することはできません。	
diagSubnet	文字列	診断サブネット名（作成済み） 例：cisco-diag-subnet	既存
insideSubnet	文字列	内部サブネット名（作成済み） 例：cisco-inside-subnet	既存
internalLbIp	文字列	内部サブネットの内部ロードバランサの IP アドレス（作成済み）。 例：1.2.3.4	既存
insideNetworkGatewayIp	文字列	内部サブネットのゲートウェイ IP アドレス（作成済み）	既存

パラメータ名	使用できる値/タイプ	説明	リソースの作成タイプ
outsideSubnet	文字列	外部サブネット名（作成済み） 例：cisco-outside-subnet	既存
outsideNetworkGatewayIp	文字列	外部サブネットゲートウェイ IP（作成済み）	既存
deviceGroupName	文字列	Firewall Management Center のデバイスグループ（作成済み）	既存
insideZoneName	文字列	Firewall Management Center の内部ゾーン名（作成済み）	既存
outsideZoneName	文字列	Firewall Management Center の外部ゾーン名（作成済み）	既存
softwareVersion	文字列	Firewall Threat Defense Virtual バージョン（展開時にドロップダウンから選択）	既存
vmSize	文字列	Firewall Threat Defense Virtual インスタンスのサイズ（展開時にドロップダウンから選択）	該当なし
ftdLicensingSku	文字列	Firewall Threat Defense Virtual ライセンスモード（PAYG/BYOL） 注：PAYG はバージョン 6.5+ でサポートされています。	該当なし
licenseCapability	カンマ区切り文字列	BASE、MALWARE、URLFilter、THREAT	該当なし
tdVmManagementUserName	文字列 *	Firewall Threat Defense Virtual VM 管理の管理者ユーザー名。 これは「admin」には**できません**。VM 管理者ユーザー名のガイドラインについては、「Azure」を参照してください。	新規作成

パラメータ名	使用できる値/タイプ	説明	リソースの作成タイプ
tdVmManagementUserPassword	文字列 *	Firewall Threat Defense Virtual VM 管理の管理者ユーザーのパスワード。 パスワードの長さは 12 ー 72 文字で、小文字、大文字、数字、特殊文字を使用する必要があります。また、文字の繰り返しは 2 回までにする必要があります。 （注） テンプレートには、このパラメータのコンプライアンスチェック機能はありません。	新規作成
ftdAdminUserPassword	文字列	Firewall Threat Defense Virtual 管理者ユーザーのパスワード。 （注） **TDvmManagementUserPassword** パラメータについて説明されている基準は、このパラメータにも適用されます。	
fmcIpAddress	文字列 x.x.x.x	Firewall Management Center のパブリック IP アドレス（作成済み）	既存
fmcUserName	文字列	管理権限を持つ Firewall Management Center ユーザー名（作成済み）	既存
fmcPassword	文字列	前述の Firewall Management Center ユーザー名の Firewall Management Center パスワード（作成済み）	既存
policyName	文字列	Firewall Management Center で作成されたセキュリティポリシー（作成済み）	既存

パラメータ名	使用できる値/タイプ	説明	リソースの作成タイプ
clusterGroupName	文字列	脅威防御 デバイスを管理センターに登録するときに使用されるクラスタ グループの名前。 例： tdv-cluster	
healthCheckPortNumber	文字列	ゲートウェイロードバランサで正常性プローブを作成するときに使用されるヘルスチェックのポート番号。 例：8080	
functionHostingPlan	文字列	機能展開のホスティング プラン（消費は消費ホスティングプランを使用、プレミアムはプレミアムホスティングプランを使用）。 デフォルト：consumption	
functionAppSubnet	文字列	関数アプリ サブネット名（作成済み）。 例： tdv-fapp-subnet	
functionAppSubnetCIDR	文字列	関数アプリ サブネットのCIDR（作成済み）。 例：10.0.4.0/24	
scalingMetricsList	文字列	スケーリングおよびスケーリングの決定に使用されるメトリック。 許可: CPU & MEMORY	

パラメータ名	使用できる値/タイプ	説明	リソースの作成タイプ
scalingPolicy	POLICY-1/POLICY-2	**POLICY-1**：設定された期間に、いずれかの Firewall Threat Defense Virtual の平均負荷がスケールアウトしきい値を超えるとスケールアウトがトリガーされます。 **POLICY-2**：設定された期間に、VMSS のすべての Firewall Threat Defense Virtual デバイスの平均負荷がスケールアウトしきい値を超えるとスケールアウトがトリガーされます。 どちらの場合も、スケールインロジックは同じままです。設定された期間に、すべての Firewall Threat Defense Virtual デバイスの平均負荷がスケールインしきい値を下回るとスケールインがトリガーされます。	該当なし
scalingMetricsList	文字列	スケーリングの決定に使用されるメトリック。 許可：CPU、MEMORY デフォルト：CPU	該当なし
cpuScaleInThreshold	文字列	CPU メトリックのスケールインしきい値（パーセント単位）。 デフォルト：10 Firewall Threat Defense Virtual メトリックがこの値を下回ると、スケールインがトリガーされます。 Azure での Firewall Threat Defense Virtual クラスタリングのオートスケール ロジック（61 ページ）を参照してください。	該当なし

パラメータ名	使用できる値/タイプ	説明	リソースの作成タイプ
cpuScaleOutThreshold	文字列	CPU メトリックのスケールアウトしきい値（パーセント単位）。 デフォルト：80 Firewall Threat Defense Virtualメトリック（CPU 使用率）がこの値を上回ると、スケールアウトがトリガーされます。 「cpuScaleOutThreshold」は、常に「cpuScaleInThreshold」より**大きく**する必要があります。 「Azure での Firewall Threat Defense Virtual クラスタリングのオートスケール ロジック（61 ページ）」を参照してください。	該当なし
memoryScaleInThreshold	文字列	メモリメトリックのスケールインしきい値（パーセント単位）。 デフォルト：0 Firewall Threat Defense Virtualメトリック（CPU 使用率）がこの値を下回ると、スケールインがトリガーされます。 「Azure での Firewall Threat Defense Virtual クラスタリングのオートスケール ロジック（61 ページ）」を参照してください。	該当なし

パラメータ名	使用できる値/タイプ	説明	リソースの作成タイプ
memoryScaleOutThreshold	文字列	メモリメトリックのスケールアウトしきい値（パーセント単位）。 デフォルト：0 Firewall Threat Defense Virtualメトリック（CPU 使用率）がこの値を上回ると、スケールアウトがトリガーされます。 「memoryScaleOutThreshold」は、常に「memoryScaleInThreshold」より**大きく**する必要があります。 「Azure での Firewall Threat Defense Virtual クラスタリングのオートスケール ロジック（61 ページ）」を参照してください。	該当なし
minFtdCount	整数	任意の時点でスケールセットで使用可能な最小 Firewall Threat Defense Virtual インスタンス数。 例：2。	該当なし
maxFtdCount	整数	スケールセットで許可される最大 Firewall Threat Defense Virtual インスタンス数。 例：10 （注） この数は Firewall Management Center の容量によって制限されます。 Auto Scale ロジックではこの変数の範囲はチェックされないため、慎重に入力してください。	該当なし

パラメータ名	使用できる値/タイプ	説明	リソースの作成タイプ
metricsAverageDuration	整数	ドロップダウンから選択します。 この数値は、メトリックが平均化される時間（分単位）を表します。 この変数の値が 5（5 分）の場合、Auto Scale Manager がスケジュールされると、メトリックの過去 5 分間の平均がチェックされ、その結果に基づいてスケーリングの判断が行われます。 （注） Azure の制限により、有効な数値は 1、5、15、および 30 だけです。	該当なし

パラメータ名	使用できる値/タイプ	説明	リソースの作成タイプ
initDeploymentMode	BULK/STEP		

パラメータ名	使用できる値/タイプ	説明	リソースの作成タイプ
		主に最初の展開、またはスケールセットに Firewall Threat Defense Virtual インスタンスが含まれていない場合に適用されます。 BULK：Auto Scale Manager は、「minFtdCount」個の Firewall Threat Defense Virtual インスタンスを同時に展開しようとします。 （注） 起動は並行して行われますが、Firewall Management Center への登録は Firewall Management Center の制限により順次実行されます。 STEP：Auto Scale Manager は、スケジュールされた間隔ごとに「minFtdCount」個の Firewall Threat Defense Virtualデバイスを 1 つずつ展開します。 （注） STEP オプションでは、「minFtdCount」個のインスタンスが Firewall Management Center で起動および設定されて、動作可能になるまで時間がかかりますが、デバッグに役立ちます。 BULK オプションでは、（並行実行のため）「minFtdCount」個すべての Firewall Threat Defense Virtual を起動するのに 1 つの Firewall Threat Defense Virtual 起動と同じ時間がかかりますが、Firewall Management Center の登録は順次実行されます。 「minFtdCount」個の Firewall Threat Defense Virtual を展開す	

パラメータ名	使用できる値/タイプ	説明	リソースの作成タイプ
		るための合計時間 =（1 つの Firewall Threat Defense Virtual の起動時間 + 1 つの Firewall Threat Defense Virtual 登録および設定時間 * minFtdCount）。	
* Azure には、新しいリソースの命名規則に関する制限があります。制限を確認するか、またはすべて小文字を使用してください**スペースやその他の特殊文字は使用しないでください**。			

Firewall Threat Defense Virtual Auto Scale の展開プロセスとリソースを備えたクラスター

Firewall Threat Defense Virtual Azure での自動スケール展開プロセスを使用した クラスターには、次の作業が含まれます。

- ARM テンプレートを展開します。
- クラスタリング機能を構築して展開します。
- 論理アプリケーションを更新して有効化します。

Azure Resource Manager テンプレート展開リソース

サンドイッチトポロジ（NLB） の ARM テンプレート（`azure_ftdv_nlb_cluster_autoscale.json`）を使用して、Azure にオートスケールを備えた Firewall Threat Defense Virtual クラスタを展開すると、リソースグループ内に次のリソースが作成されます。

- 仮想マシンスケールセット（VMSS）
- 外部ロードバランサ
- 内部ロードバランサ
- Azure Function App
- Logic App
- セキュリティグループ（データインターフェイスおよび管理インターフェイス用）

GWLB の ARM テンプレート（`azure_ftdv_gwlb_cluster_autoscale.json`）を使用して、Azure にオートスケールを備えた Firewall Threat Defense Virtual クラスタを展開すると、次のリソースがリソースグループ内に作成されます。

- 仮想マシン（VM）または仮想マシンスケールセット（VMSS）
- ゲートウェイロードバランサ（GWLB）

- Azure Function App
- Logic App
- ネットワーキング インフラストラクチャ
- 展開に必要なセキュリティグループおよびその他のコンポーネント。

オートスケールソリューションを使用するFirewallThreatDefenseVirtualクラスタの展開

ARM テンプレートを使用して、Azure に Auto Scale ソリューションを使用する Threat Defense Virtual クラスタリングを展開します。トポロジ、サンドイッチ（NLB）または GWLB のユースケースに基づいて、Azure の自動スケールソリューションを使用して Firewall Threat Defense Virtual クラスタリングを展開するための適切な ARM テンプレートをダウンロードして構成する必要があります。

始める前に

GitHub から展開パッケージをダウンロードする

Azure 向けの NLB ソリューションを使用する Firewall Threat Defense Virtual クラスタリング オートスケールは、Azure Resource Manager（ARM）テンプレートベースの展開であり、Azure が提供するサーバーレスインフラストラクチャ（論理アプリ、Azure 関数、ロードバランサ、仮想マシンスケールセットなど）を使用します。

Azure 向けの GWLB ソリューションを使用する Firewall Threat Defense Virtual クラスタリング オートスケールは、ARM テンプレートベースの展開であり、GWLB、ネットワーク インフラストラクチャ、脅威防御仮想オート スケーリング グループ、サーバーレスコンポーネント、および他の必要なリソースを作成します。

両方のソリューションの展開手順はほぼ同じです。

Azure 向けのオートスケール ソリューションを使用する Firewall Threat Defense Virtual クラスタリングの起動に必要なファイルをダウンロードします。

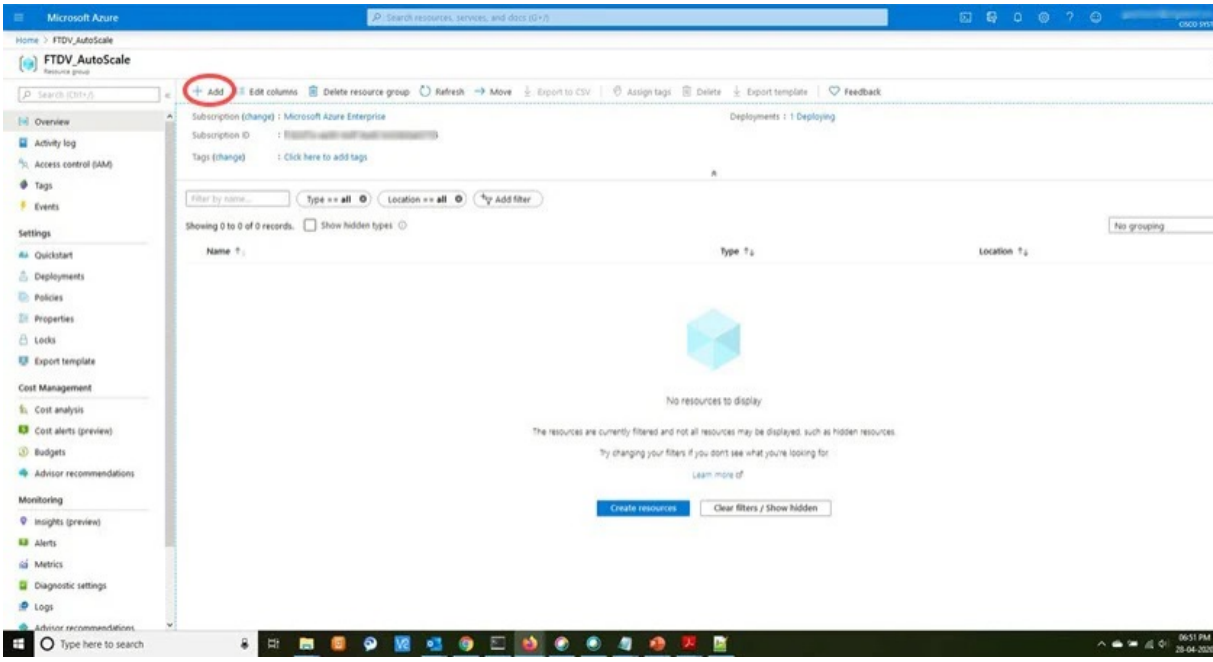
該当するバージョン用の展開スクリプトとテンプレートは、GitHub リポジトリから入手できます。

手順

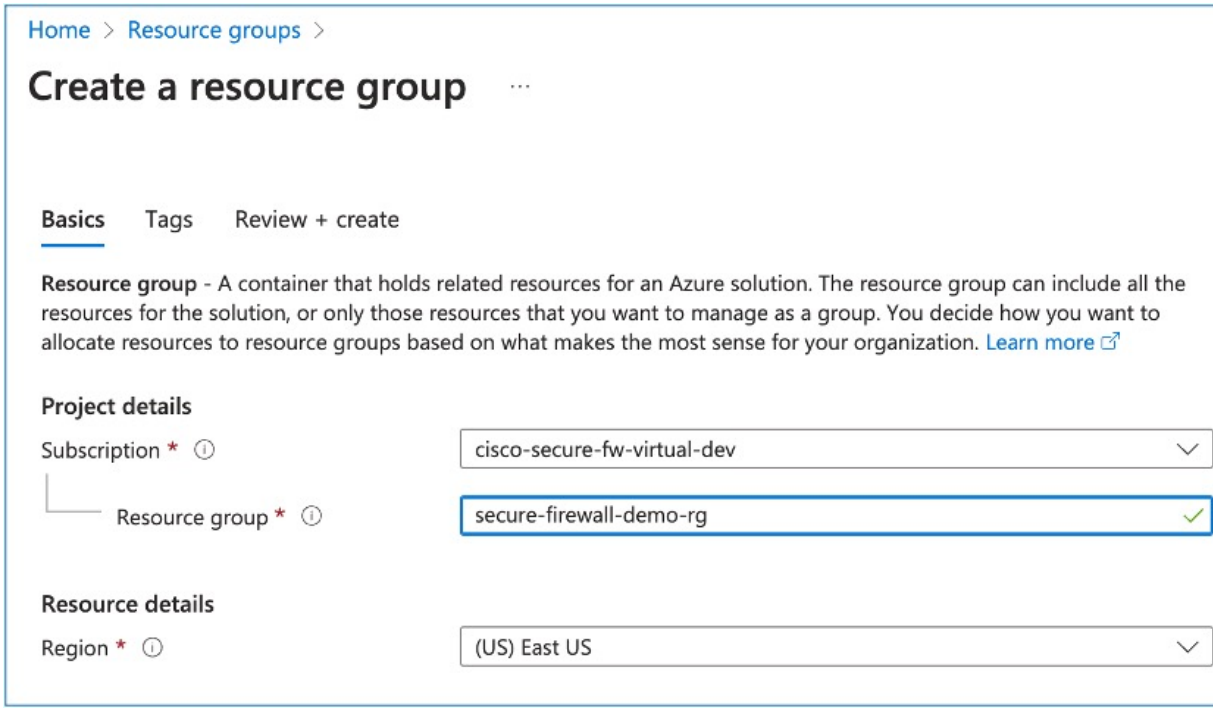
ステップ 1 Microsoft アカウントのユーザー名とパスワードを使用して、Microsoft Azure ポータル（https://portal.azure.com）にログインします。

ステップ 2 **[リソースグループ（Resource Groups）]** ブレードにアクセスするには、サービスのメニューから **[リソースグループ（Resource groups）]** をクリックします。サブスクリプション内のすべてのリソースグループがブレードに一覧表示されます。新しいリソース グループを作成する

か、既存の空のリソース グループを選択します。たとえば、 **threat Defense virtual_AutoScale** です。

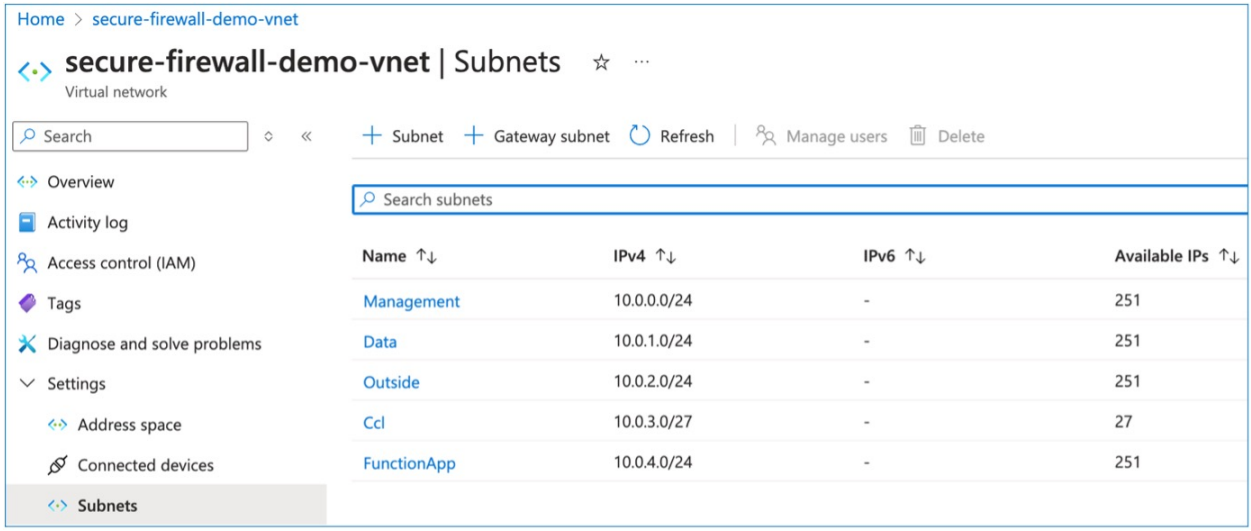


ステップ 3 [リソースの作成（+）（Create a resource (+)）] をクリックして、テンプレート展開用の新しいリソースを作成します。**[リソース グループの作成（Create Resource Group）]** ブレードが表示されます。



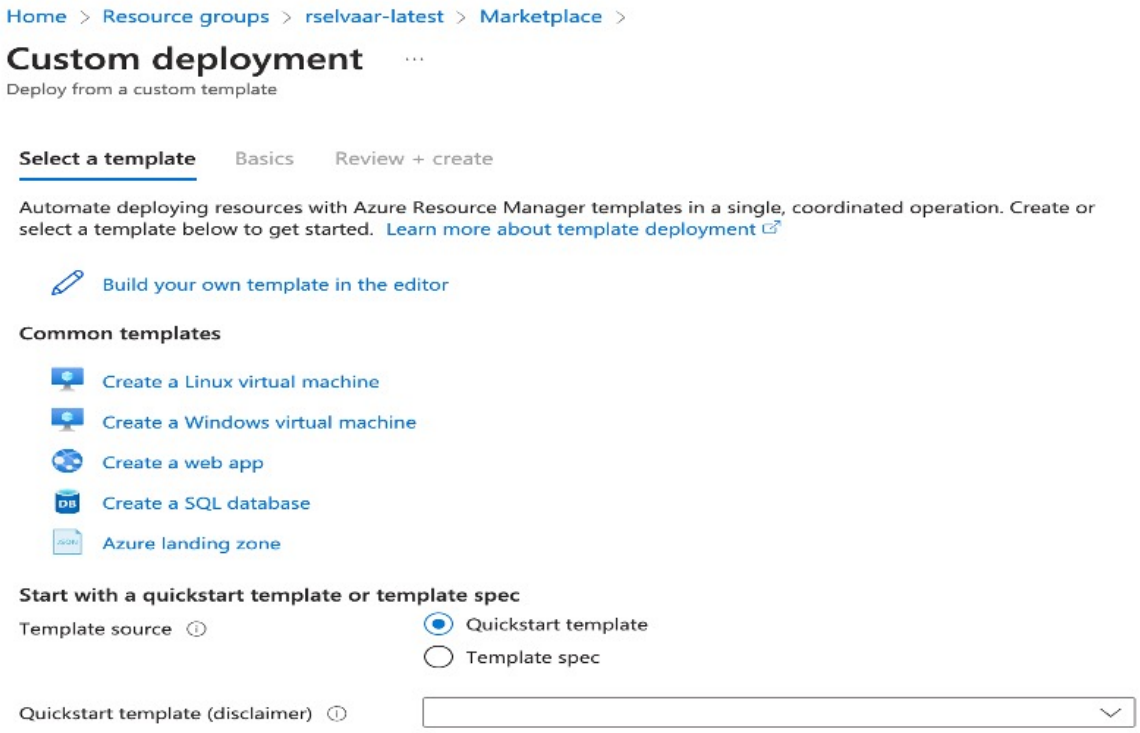
ステップ 4 4. サービスのメニューから **[仮想ネットワーク（Virtual Network）]** をクリックして、[仮想ネットワーク（Virtual network）] ブレードにアクセスします。サブネットを含む仮想ネットワークを作成します。

- GWLB 展開の場合、管理、データ、CCL サブネット、および関数アプリを持つ仮想ネットワークを作成します。
- NLB 展開の場合、管理、内部、外部、CCL のサブネットおよび関数アプリを持つ仮想ネットワークを作成します。

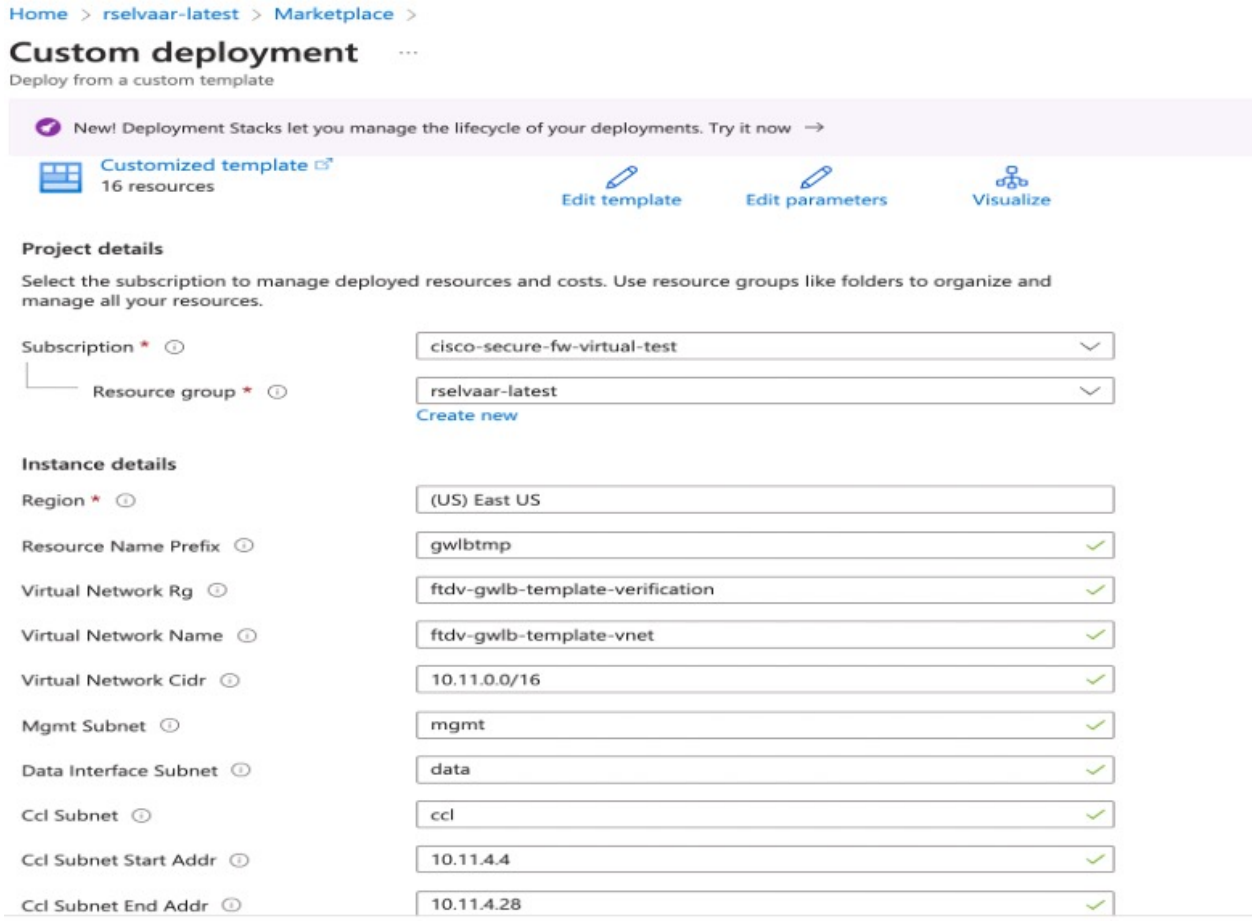


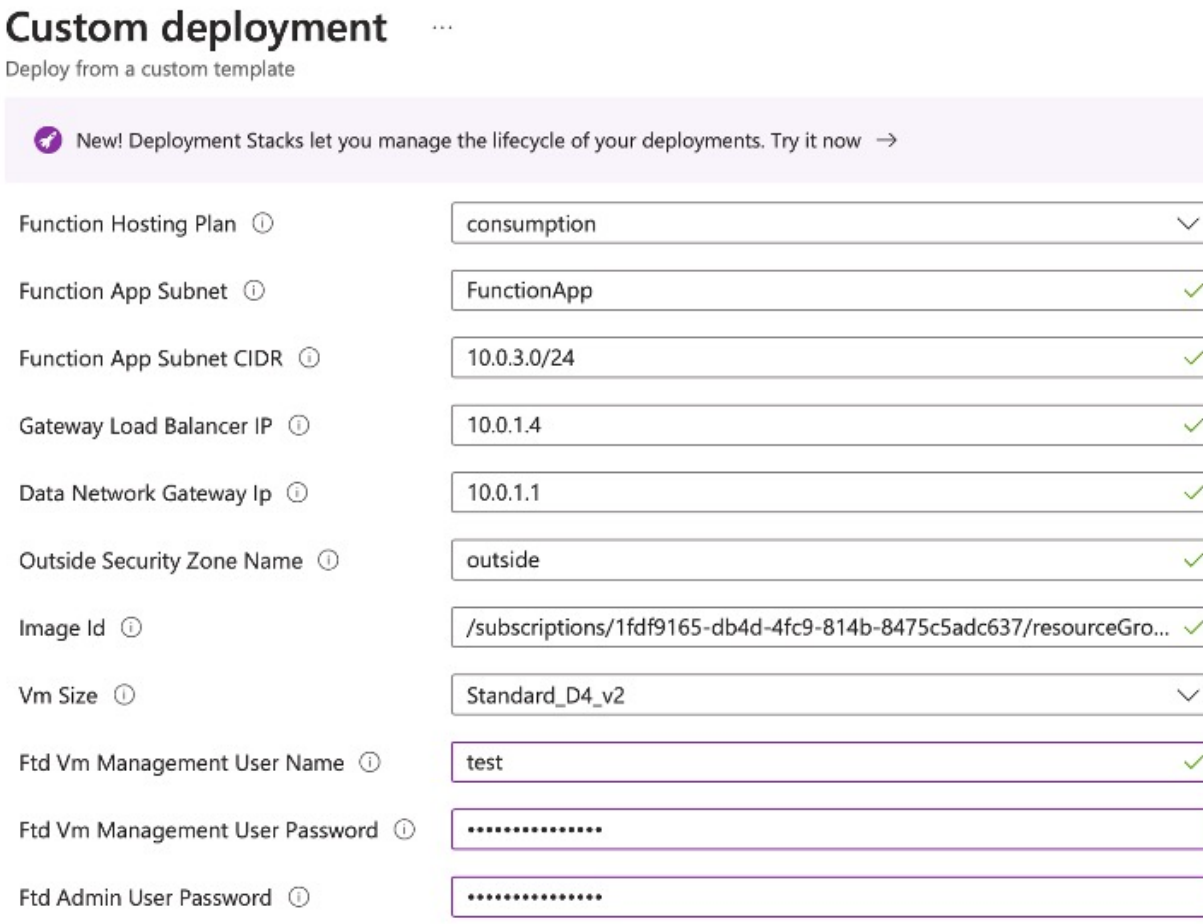
ステップ 5 **[マーケットプレイスの検索（Search the Marketplace）]** で、**[テンプレート展開（Template deployment）]**（カスタム テンプレートを使用した展開）と入力し、**Enter** を押します。

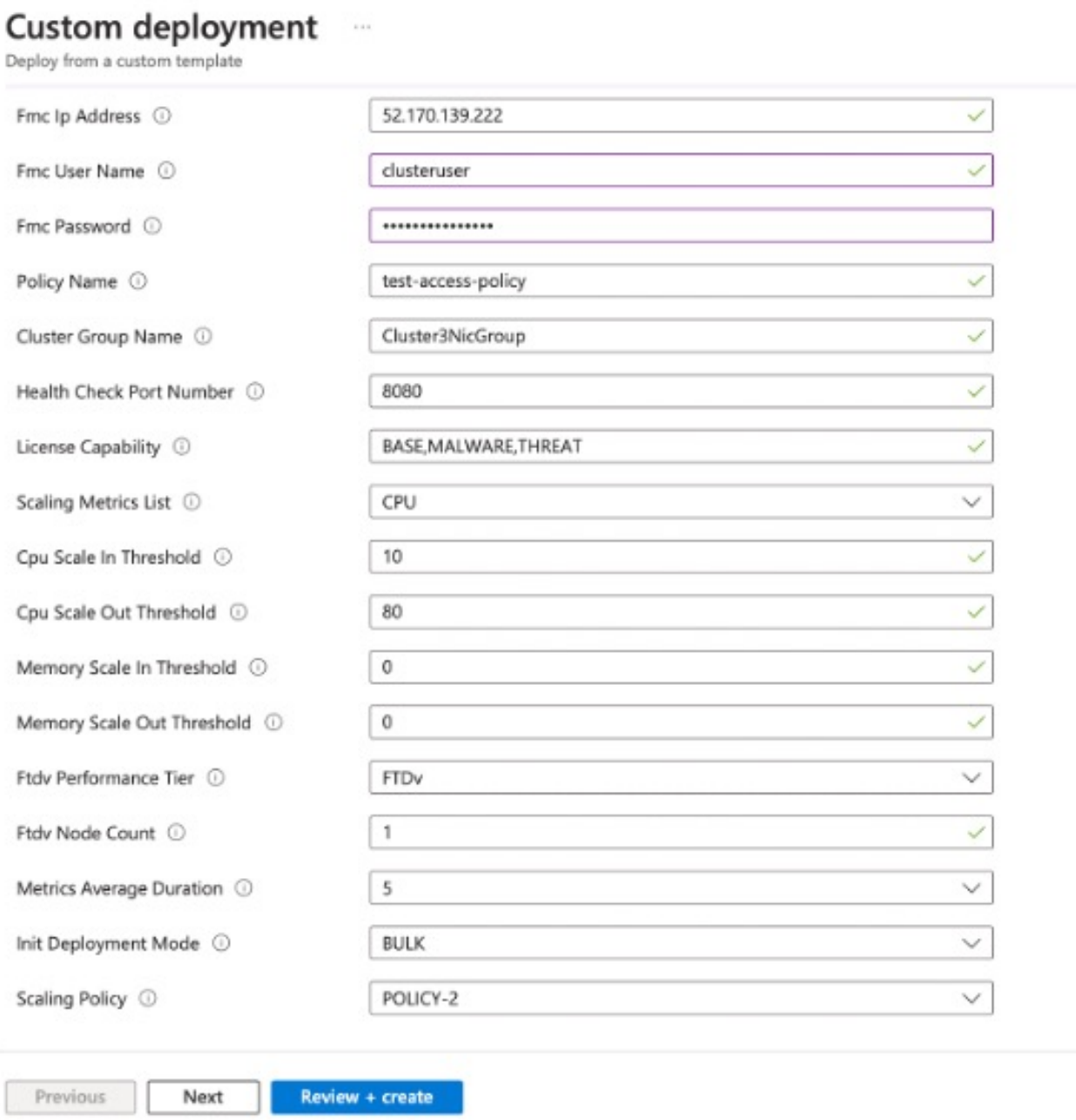
ステップ 6 [作成（Create）] をクリックします。テンプレートを作成するためのオプションは複数あります。[エディタで独自のテンプレートを作成する（Build your own template in editor）] を選択します。



ステップ 7 **[テンプレートの編集（Edit template）]** ウィンドウで、すべてのデフォルトコンテンツを削除し、更新した `azure_ftdv_gwlb_cluster_custom_image.json` または `azure_ftdv_nlb_cluster_custom_image.json` から（Azure に展開するオートスケール ソリューションのタイプに応じて）コンテンツをコピーして、**[保存（Save）]** をクリックします。または、**[ファイルの読み込み（Load file）]** をクリックし、コンピュータからこのファイルを参照してアップロードします。


Home > rselvaar-latest > Marketplace >
Custom deployment
Deploy from a custom template
New! Deployment Stacks let you manage the lifecycle of your deployments. Try it now →
Customized template
16 resources
Edit template
Edit parameters
Visualize
Project details
Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.
Subscription *
cisco-secure-fw-virtual-test
Resource group *
rselvaar-latest
Create new
Instance details
Region *
(US) East US
Resource Name Prefix
gwlbtmp
Virtual Network Rg
ftdv-gwlb-template-verification
Virtual Network Name
ftdv-gwlb-template-vnet
Virtual Network Cidr
10.11.0.0/16
Mgmt Subnet
mgmt
Data Interface Subnet
data
Ccl Subnet
ccl
Ccl Subnet Start Addr
10.11.4.4
Ccl Subnet End Addr
10.11.4.28


Custom deployment
Deploy from a custom template
New! Deployment Stacks let you manage the lifecycle of your deployments. Try it now →
Function Hosting Plan
consumption
Function App Subnet
FunctionApp
Function App Subnet CIDR
10.0.3.0/24
Gateway Load Balancer IP
10.0.1.4
Data Network Gateway Ip
10.0.1.1
Outside Security Zone Name
outside
Image Id
/subscriptions/1fdf9165-db4d-4fc9-814b-8475c5adc637/resourceGro...
Vm Size
Standard_D4_v2
Ftd Vm Management User Name
test
Ftd Vm Management User Password
Ftd Admin User Password



ステップ 8 パラメータ フィールド セクションで、すべてのパラメータを入力します。各パラメータの詳細については、「入力パラメータ」を参照してください。次に、**[レビューと作成（Review+Create）]** をクリックします。

ステップ 9 テンプレートの展開が成功すると、Azure 向けの脅威防御仮想オートスケール ソリューションに必要なすべてのリソースが作成されます。次の図のリソースを参照してください。**[タイプ（Type）]** 列には、論理アプリ、VMSS、ロードバランサ、パブリック IP アドレスなどの各リソースが示されます。

次のタスク

Azure 関数アプリの展開 （83 ページ）。

Azure 関数アプリの展開

ARM テンプレートを展開すると、Azure は次の名前で関数アプリを作成します：**<resourceNamePrefix>-function-app**。

手順

ステップ 1 ARM テンプレートを展開したときに作成した関数アプリに移動し、次の手順を実行します。

ローカルコンピュータから次のコマンドを実行して、クラスタ自動スケール Azure 関数を関数アプリに展開します。

```
az functionapp deployment source config-zip -g <Resource Group Name>
-n <Function App Name> --src  <cluster_functions.zip> --build-remote true
```

ステップ 2 Azure 関数の展開後、関数アプリケーションの概要セクションにアップロードされた関数を表示できます。

Update the Azure Logic App

The Logic App acts as the orchestrator for the Autoscale functionality. The ARM template creates a skeleton Logic App, which you then need to update manually to provide the information necessary to function as the auto scale orchestrator.

手順

ステップ 1 From the repository, retrieve the file *LogicApp.txt* to the local system and edit as shown below.

重要
Read and understand all of these steps before proceeding.

These manual steps are not automated in the ARM template so that only the Logic App can be upgraded independently later in time.

a) 必須: Find and replace all the occurrences of “SUBSCRIPTION_ID” with your subscription ID information.
b) 必須: Find and replace all the occurrences of “RG_NAME” with your resource group name.
c) 必須: Find and replace all of the occurrences of “FUNCTIONAPPNAME” to your function app name.

The following example shows a few of these lines in the *LogicApp.txt* file:

```
"AutoScaleManager": {
```

```
      "inputs": {
          "function": {
              "id":
"/subscriptions/SUBSCRIPTION_ID/resourceGroups/RG_NAME/providers/Microsoft.Web/sites/FUNCTIONAPPNAME/functions/AutoScaleManager"

          }
.
.
                      },
                      "Deploy_Changes_to_FTD": {
                          "inputs": {
                              "body": "@body('AutoScaleManager')",
                              "function": {
                                  "id":
"/subscriptions/SUBSCRIPTION_ID/resourceGroups/RG_NAME/providers/Microsoft.Web/sites/FUNCTIONAPPNAME/functions/DeployConfiguration"

                              }
.
.
                      "DeviceDeRegister": {
                          "inputs": {
                              "body": "@body('AutoScaleManager')",
                              "function": {
                                  "id":
"/subscriptions/SUBSCRIPTION_ID/resourceGroups/RG_NAME/providers/Microsoft.Web/sites/FUNCTIONAPPNAME/functions/DeviceDeRegister"

                              }
                          },
                          "runAfter": {
                              "Delay_For_connection_Draining": [
```

d) （任意） Edit the trigger interval, or leave the default value (5). This is the time interval at which the Autoscale functionality is periodically triggered. The following example shows these lines in the *LogicApp.txt* file:

```
    "triggers": {
        "Recurrence": {
            "conditions": [],
            "inputs": {},
            "recurrence": {
                "frequency": "Minute",
                "interval": 5
            },
```

e) （任意） Edit the time to drain, or leave the default value (5). This is the time interval to drain existing connections from the Firewall Threat Defense Virtual before deleting the device during the Scale-In operation. The following example shows these lines in the *LogicApp.txt* file:

```
     "actions": {
          "Branch_based_on_Scale-In_or_Scale-Out_condition": {
              "actions": {
                  "Delay_For_connection_Draining": {
                      "inputs": {
                          "interval": {
                              "count": 5,
                              "unit": "Minute"
                          }
```

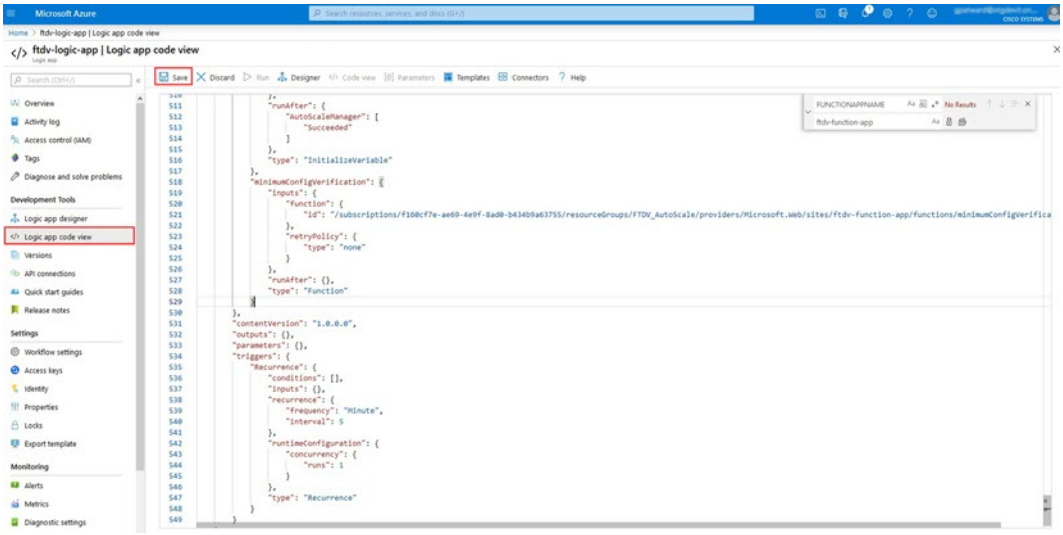
f) （任意） Edit the cool down time, or leave the default value (10). This is the time to perform NO ACTION after the Scale-Out is complete. The following example shows these lines in the *LogicApp.txt* file:

```
"actions": {
    "Branch_based_on_Scale-Out_or_Invalid_condition": {
        "actions": {
            "Cooldown_time": {
                "inputs": {
                    "interval": {
                        "count": 10,
                        "unit": "Second"
                    }
```

（注）
These steps can also be done from the Azure portal. Consult the Azure documentation for more information.

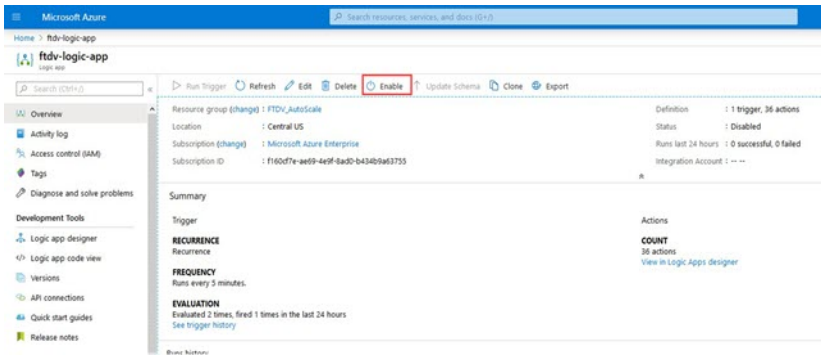
ステップ 2 Go to the **Logic App code view**, delete the default contents and paste the contents from the edited *LogicApp.txt* file, and click **Save**.

図 *15 : Logic App Code View*



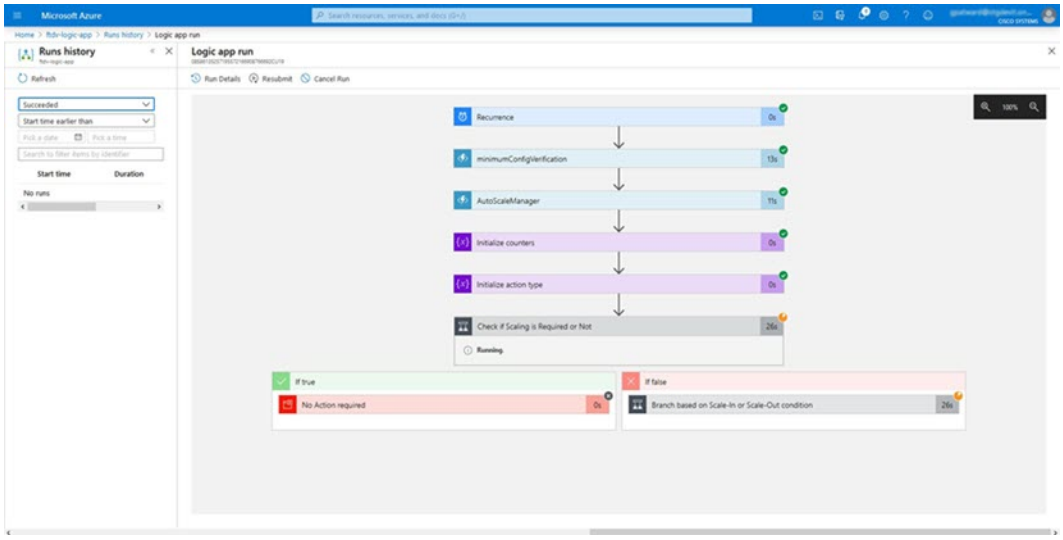
ステップ 3 When you save the Logic App, it is in a 'Disabled' state. Click **Enable** when you want to start the Auto Scale Manager.

図 *16 : Enable Logic App*



ステップ **4** Once enabled, the tasks start running. Click the 'Running' status to see the activity.

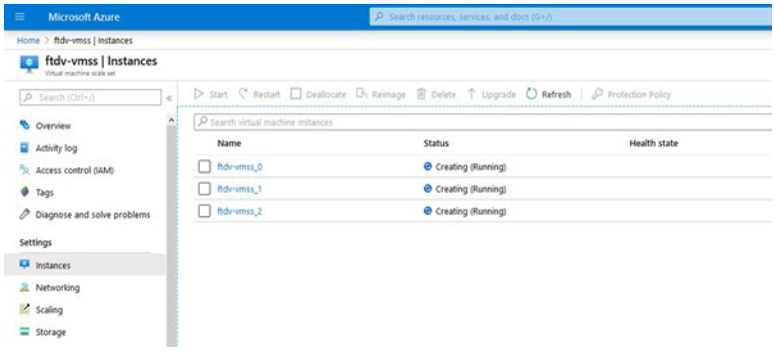
図 *17 : Logic App Running Status*



ステップ **5** Once the Logic App starts, all the deployment-related steps are complete.

ステップ **6** Verify in the VMSS that Firewall Threat Defense Virtual instances are being created.

図 *18 : Threat Defense Virtual Instances Running*



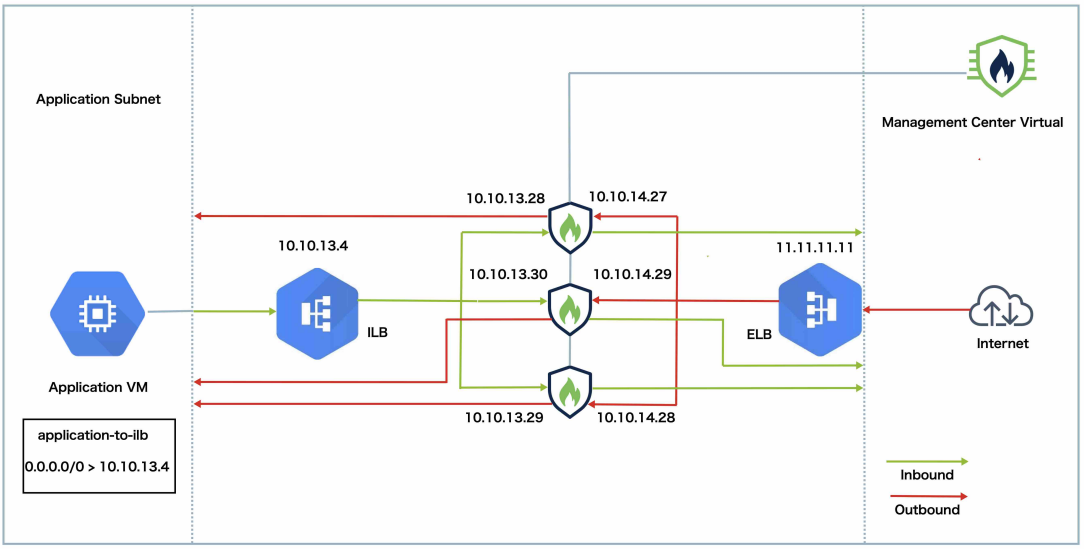
In this example, three Firewall Threat Defense Virtual instances are launched because 'minFtdCount' was set to '3' and 'initDeploymentMode' was set to 'BULK' in the ARM template deployment.

GCP でのクラスタの展開

クラスタを GCP で展開するには、手動で展開するか、インスタンステンプレートを使用してインスタンスグループを展開します。GCP ロードバランサ、または Cisco Cloud Services Router などの非ネイティブのロードバランサでクラスタを使用できます。

（注） 発信トラフィックはインターフェイス NAT が必要であり、64K 接続に制限されています。

トポロジの例



このトポロジは、着信と発信の両方のトラフィックフローを示しています。Threat Defense Virtual クラスタは、内部ロードバランサと外部ロードバランサの間に挟まれています。Management Center Virtual インスタンスは、クラスタの管理に使用されます。

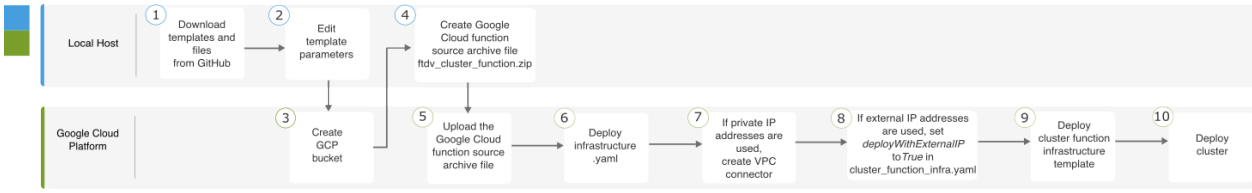
インターネットからの着信トラフィックは、外部ロードバランサに送られ、そこから Threat Defense Virtual クラスタにトラフィックが送信されます。トラフィックは、クラスタ内の Threat Defense Virtual インスタンスによって検査された後、アプリケーション VM に転送されます。

アプリケーション VM からの発信トラフィックは、内部ロードバランサに送信されます。その後、トラフィックは Threat Defense Virtual クラスタに転送され、インターネットに送信されます。

GCP で Threat Defense Virtual クラスタを展開するためのエンドツーエンドのプロセス

テンプレートベースの展開

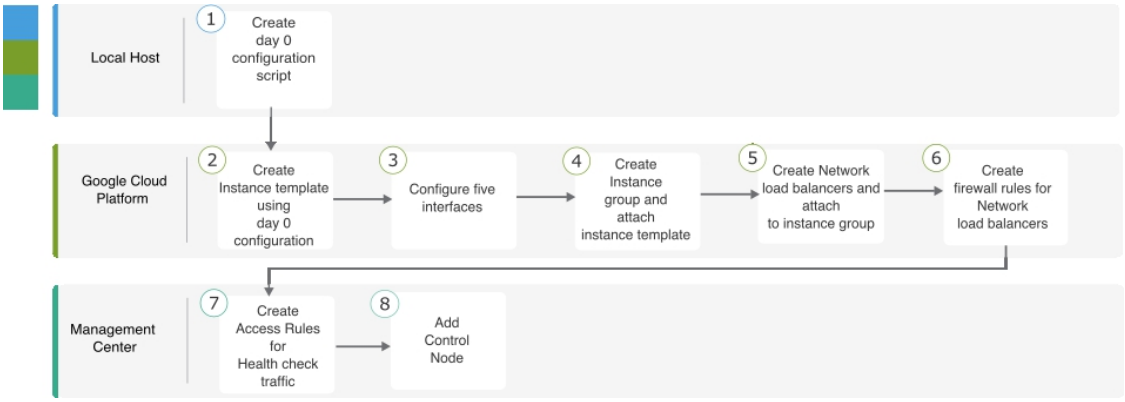
次のフローチャートは、GCP での Threat Defense Virtual クラスタのテンプレートベース展開のワークフローを示しています。



	ワークスペース	手順
1	ローカルホスト	GitHub からテンプレートとファイルをダウンロードします。
2	ローカルホスト	テンプレートパラメータを編集します。
3	Google Cloud Platform	GCP バケットを作成します。
4	ローカルホスト	Google Cloud 関数ソースアーカイブファイル ftdv_cluster_function.zip を作成します。
5	Google Cloud Platform	Google 関数ソースアーカイブファイルをアップロードします。
6	Google Cloud Platform	infrastructure.yaml を展開します。
7	Google Cloud Platform	プライベート IP アドレスが使用されている場合は、VPC コネクタを作成します。
8	Google Cloud Platform	外部 IP アドレスが使用されている場合は、cluster_function_infra.yaml で deployWithExternalIP を True に設定します。
9	Google Cloud Platform	クラスタ機能インフラストラクチャ テンプレートを展開します。
10	Google Cloud Platform	クラスタを展開します。

手動展開

次のフローチャートは、GCP での Threat Defense Virtual クラスタの手動展開のワークフローを示しています。



	ワークスペース	手順
1	ローカルホスト	Day 0 構成スクリプトを作成します。
2	Google Cloud Platform	Day 0 構成を使用してインスタンステンプレートを作成します。
3	Google Cloud Platform	インターフェイスを設定します。
4	Google Cloud Platform	インスタンスグループを作成し、インスタンステンプレートを割り当てます。
5	Google Cloud Platform	NLB を作成し、インスタンスグループにアタッチします。
6	Google Cloud Platform	NLB のファイアウォールルールを作成します。
7	Management Center	ヘルスチェックトラフィックのアクセスルールを作成します。
8	Management Center	制御ノードを追加します。

テンプレート

以下のテンプレートは GitHub で入手できます。パラメータ値は、テンプレートで指定されたパラメータ名、および値であり、自明です。

- East-West トラフィック用のクラスタ展開テンプレート：deploy_ngfw_cluster_yaml
- North-South トラフィック用のクラスタ展開テンプレート：deploy_ngfw_cluster.yaml

インスタンステンプレートを使用したGCPでのインスタンスグループの展開

インスタンステンプレートを使用して、GCP にインスタンスグループを展開します。

始める前に

- 展開には Google Cloud Shell を使用します。または、任意の macOS/Linux/Windows マシンで Google SDK を使用できます。
- クラスタが Management Center に自動登録されるようにするには、REST API を使用できる管理者権限を持つユーザーを Management Center で作成する必要があります。*Cisco Secure Firewall Management Center Administration Guide*を参照してください。
- cluster_function_infra.yaml で指定したポリシー名と一致するアクセスポリシーを Management Center に追加します。

手順

ステップ 1 テンプレートを GitHub からローカルフォルダにダウンロードします。

ステップ 2 必要な resourceNamePrefix パラメータ（ngfwvcls など）と他の必要なユーザー入力を使用して、**infrastructure.yaml**、**cluster_function_infra.yaml**、および **deploy_ngfw_cluster.yaml** を編集します。

Cisco Secure Firewall バージョン 7.4.1 以降では、診断インターフェイスを使用せずにクラスタを展開できます。外部、内部、管理、および CCL インターフェイスのみを使用してクラスタを展開するには、**infrastructure.yaml** ファイルと **deploy_ngfw_cluster.yaml** ファイルの両方で *withDiagnostic* 変数を **False** に設定します。

deploy_ngfw_cluster.yaml ファイルは、GitHub で **east-west** フォルダと **north-south** フォルダの両方にあることに注意してください。トラフィックフローの要件に従って、適切なテンプレートをダウンロードします。

ステップ 3 Google Cloud Shell を使用してバケットを作成し、Google Cloud 関数ソースアーカイブファイル ftdv_cluster_function.zip をアップロードします。

gsutil mb --pap enforced gs://*resourceNamePrefix*-ftdv-cluster-bucket/

ここでの *resourceNamePrefix* 変数が **cluster_function_infra.yaml** で指定した resourceNamePrefix 変数と一致していることを確認します。

ステップ 4 クラスタ インフラストラクチャのアーカイブファイルを作成します。

例：

```
zip -j ftdv_cluster_function.zip ./cluster-function/*
```

ステップ 5 前に作成した Google ソースアーカイブをアップロードします。

gsutil cp ftdv_cluster_function.zip gs://*resourceNamePrefix***-ftdv-cluster-bucket/**

ステップ 6 クラスタのインフラストラクチャを展開します。

gcloud deployment-manager deployments create *cluster_name* **--config infrastructure.yaml**

ステップ 7 プライベート IP アドレスを使用している場合は、次の手順を実行します。

a) Threat Defense Virtual 管理 VPC を使用して、Management Center Virtual を起動してセットアップします。

b) VPC コネクタを作成して、Google Cloud 関数を Threat Defense Virtual 管理 VPC に接続します。

gcloud compute networks vpc-access connectors create *vpc-connector-name* **--region us-central1 --subnet** *resourceNamePrefix***-ftdv-mgmt-subnet28**

ステップ 8 Management Center が Threat Defense Virtual からリモートに配置され、Threat Defense Virtual に外部 IP アドレスが必要な場合は、必ず **cluster_function_infra.yaml** で **deployWithExternalIP** を **True** に設定してください。

ステップ 9 クラスタ機能インフラストラクチャを展開します。

gcloud deployment-manager deployments create *cluster_name* **--config cluster_function_infra.yaml**

ステップ 10 クラスタを展開します。

1. North-South トポロジ展開の場合：

   gcloud deployment-manager deployments create *cluster_name* **--config north-south/deploy_ngfw_cluster.yaml**

2. East-West トポロジ展開の場合：

   gcloud deployment-manager deployments create *cluster_name* **--config east-west/deploy_ngfw_cluster.yaml**

GCP でのクラスタの手動展開

クラスタを手動で展開するには、Day0 構成を準備し、各ノードを展開してから制御ノードを Firewall Management Center に追加します。

GCP 向け Day 0 構成の作成

固定構成またはカスタマイズ構成のいずれかを使用できます。

GCP 向け固定構成を使用した Day 0 構成の作成

固定構成により、クラスタのブートストラップ構成が自動生成されます。

```
{
    "AdminPassword": "password",
    "Hostname": "hostname",
```

```
    "FirewallMode": "Routed",
    "ManageLocally": "No",
                "Diagnostic": "OFF",       //Optional user input from version 7.4.1 -
use to deploy cluster without Diagnostic interface
    "Cluster": {
        "CclSubnetRange": "ip_address_start ip_address_end",
        "ClusterGroupName": "cluster_name"
    }
}
```

次に例を示します。

```
{
    "AdminPassword": "DeanW1nche$ter",
    "Hostname": "ciscoftdv",
    "FirewallMode": "Routed",
    "ManageLocally": "No",
    "Cluster": {
        "CclSubnetRange": "10.10.55.2 10.10.55.253",     //mandatory user input
        "ClusterGroupName": "ftdv-cluster"               //mandatory user input
    }
}
```

（注） 上記の設定をコピーして貼り付ける場合は、設定から **//mandatory user input** を必ず削除してください。

CclSubnetRange 変数では、サブネット内の最初の 2 つの IP アドレスと最後の 2 つの IP アドレスを使用できないことに注意してください。詳細については、「Reserved IP addresses in IPv4 subnets」を参照してください。クラスタリングに使用可能な IP アドレスが 16 個以上あることを確認します。開始 IP アドレスと終了 IP アドレスの例を次に示します。

表 *5*：開始 *IP* アドレスと終了 *IP* アドレスの例

CIDR	開始 IP アドレス	終了 IP アドレス
10.1.1.0/27	10.1.1.2	10.1.1.29
10.1.1.32/27	10.1.1.34	10.1.1.61
10.1.1.64/27	10.1.1.66	10.1.1.93
10.1.1.96/27	10.1.1.98	10.1.1.125
10.1.1.128/27	10.1.1.130	10.1.1.157
10.1.1.160/27	10.1.1.162	10.1.1.189
10.1.1.192/27	10.1.1.194	10.1.1.221
10.1.1.224/27	10.1.1.226	10.1.1.253
10.1.1.0/24	10.1.1.2	10.1.1.253

GCP 向けカスタマイズ構成を使用した Day 0 構成の作成

コマンドを使用して、クラスタのブートストラップ設定をすべて入力できます。

```
{
    "AdminPassword": "password",
    "Hostname": "hostname",
    "FirewallMode": "Routed",
    "ManageLocally": "No",
    "run_config": [comma_separated_threat_defense_configuration]
    }
```

次の例では、管理、内部、および外部インターフェイスと、クラスタ制御リンク用の VXLAN インターフェイスを使用して構成を作成します。太字の値はノードごとに一意である必要があることに注意してください。

```
{
 "AdminPassword": "W1nch3sterBr0s",
 "Hostname": "ftdv1",
 "FirewallMode": "Routed",
 "ManageLocally": "No",
 "run_config": [
  "cluster interface-mode individual force",
  "interface Management0/0",
  "management-only",
  "nameif management",
  "ip address dhcp",
  "interface GigabitEthernet0/0",
  "no shutdown",
  "nameif outside",
  "ip address dhcp",
  "interface GigabitEthernet0/1",
  "no shutdown",
  "nameif inside",
  "ip address dhcp",
  "interface GigabitEthernet0/2",
  "nve-only cluster",
  "nameif ccl_link",
  "ip address dhcp",
  "no shutdown",
  "interface vni1",
  "description Clustering Interface",
  "segment-id 1",
  "vtep-nve 1",
  "object network ccl#link",
  "range 10.1.90.2 10.1.90.17",
  "object-group network cluster#group",
  "network-object object ccl#link",
  "nve 1",
  "encapsulation vxlan",
  "source-interface ccl_link",
  "peer-group cluster#group",
  "cluster group ftdv-cluster",
  "local-unit 1",
  "cluster-interface vni1 ip 10.1.1.1 255.255.255.0",
  "priority 1",
  "enable",
  "mtu outside 1400",
  "mtu inside 1400"
 ]
}
```

（注） クラスタ制御リンク ネットワーク オブジェクトには、アドレスを必要な数だけ指定します（最大 16個）。範囲を大きくすると、パフォーマンスに影響する可能性があります。

クラスタノードの手動展開

クラスタが形成されるようにクラスタノードを展開します。GCP でのクラスタリングの場合、4 vCPU マシンタイプは使用できません。4 vCPU マシンタイプがサポートするインターフェイスは 4 つのみですが、インターフェイスは 5 つ必要です。c2-standard-8 など、5 つのインターフェイスがサポートされるマシンタイプを使用します。

手順

ステップ 1 5 つのインターフェイス（外部、内部、管理、診断、クラスタ制御リンク）を備えた Day 0 構成を使用して、インスタンステンプレートを作成します（[メタデータ（Metadata）] > [スタートアップスクリプト（Startup Script）] セクション）。

Secure Firewall Threat Defense Virtual getting started guides を参照してください。

ステップ 2 インスタンスグループを作成し、インスタンステンプレートを割り当てます。

ステップ 3 GCP ネットワークロードバランサ（内部および外部）を作成し、インスタンスグループを割り当てます。

ステップ 4 GCP ネットワークロードバランサの場合、Management Center のセキュリティポリシーでヘルスチェックを許可します。GCP ネットワークロードバランサのヘルスチェックの許可 （94 ページ） を参照してください。

ステップ 5 Management Center に制御ノードを追加します。「Management Center へのクラスタの追加（手動展開） （96 ページ）」を参照してください。

GCP ネットワークロードバランサのヘルスチェックの許可

Google Cloud は、バックエンドがトラフィックに応答するかどうかを判断するヘルスチェック機能を提供します。

ネットワークロードバランサのファイアウォール ルールを作成するには、「https://cloud.google.com/load-balancing/docs/health-checks」を参照してください。次に、Firewall Management Center でヘルスチェックトラフィックを許可するアクセスルールを作成します。必要なネットワーク範囲については、「https://cloud.google.com/load-balancing/docs/health-check-concepts」を参照してください。 アクセス コントロール ルールを参照してください。

また、動的な手動 NAT ルールを設定して、ヘルスチェックトラフィックを 169.254.169.254 の Google メタデータサーバーにリダイレクトする必要もあります。 ダイナミック手動 NAT の設定を参照してください。

正常性プローブの構成に使用されるすべてのインターフェイスに対する GCP 正常性チェックのルートを設定できます。これは、GCP正常性チェック用のルートがまだ利用可能になっていないインターフェイスで、より高いメトリックを持つルートを作成することで実現できます。

North-South NAT ルールの設定例

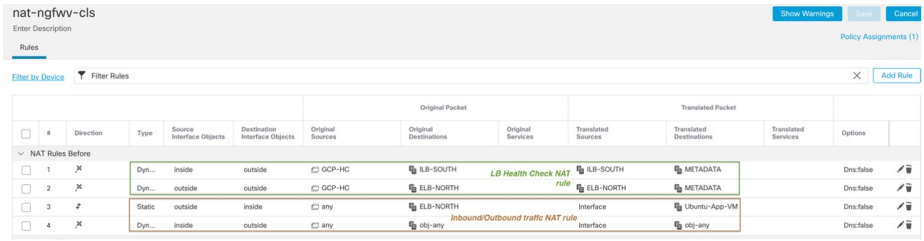
```
nat (inside,outside) source dynamic GCP-HC ILB-SOUTH destination static ILB-SOUTH METADATA
nat (outside,outside) source dynamic GCP-HC ELB-NORTH destination static ELB-NORTH
METADATA

nat (outside,inside) source static any interface destination static ELB-NORTH Ubuntu-App-VM
nat (inside,outside) source dynamic any interface destination static obj-any obj-any

object network Metadata
 host 169.254.169.254

object network ILB-SOUTH
host <ILB_IP>
object network ELB-NORTH
host <ELB_IP>

object-group network GCP-HC
 network-object 35.191.0.0 255.255.0.0
 network-object 130.211.0.0 255.255.252.0
 network-object 209.85.204.0 255.255.252.0
 network-object 209.85.152.0 255.255.252.0
```



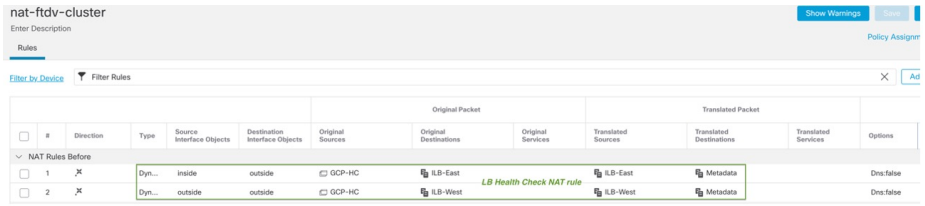
East-West NAT ルールの設定例

```
nat (inside,outside) source dynamic GCP-HC ILB-East destination static ILB-East Metadata
nat (outside,outside) source dynamic GCP-HC ILB-West destination static ILB-West Metadata

object network Metadata
 host 169.254.169.254

object network ILB-East
host <ILB_East_IP>
object network ILB-West
host <ILB_West_IP>

object-group network GCP-HC
 network-object 35.191.0.0 255.255.0.0
 network-object 130.211.0.0 255.255.252.0
 network-object 209.85.204.0 255.255.252.0
 network-object 209.85.152.0 255.255.252.0
```



ノースサウスおよびイーストウェスト トラフィック ルーティングの設定例

```
route outside 0.0.0.0 0.0.0.0 <Outside_Gateway> 1
route inside 35.191.0.0 255.255.0.0 <Inside_Gateway> 1
route inside 130.211.0.0 255.255.252.0 <Inside_Gateway> 1
route inside 209.85.152.0 255.255.252.0 <Inside_Gateway> 1
route inside 209.85.204.0 255.255.252.0 <Inside_Gateway> 1
```

デフォルト ルートが使用できない場合、ヘルス チェック用のトラフィックのルーティングにポリシーベース ルーティングを使用できます。

Management Center へのクラスタの追加（手動展開）

クラスタを手動で展開した場合は、この手順を使用してクラスタを Firewall Management Center に追加します。テンプレートを使用した場合、クラスタは自動的に Firewall Management Center に登録されます。

クラスタ ユニットのいずれかを新しいデバイスとして Firewall Management Center に追加します。Firewall Management Center は、他のすべてのクラスタ メンバーを自動検出します。

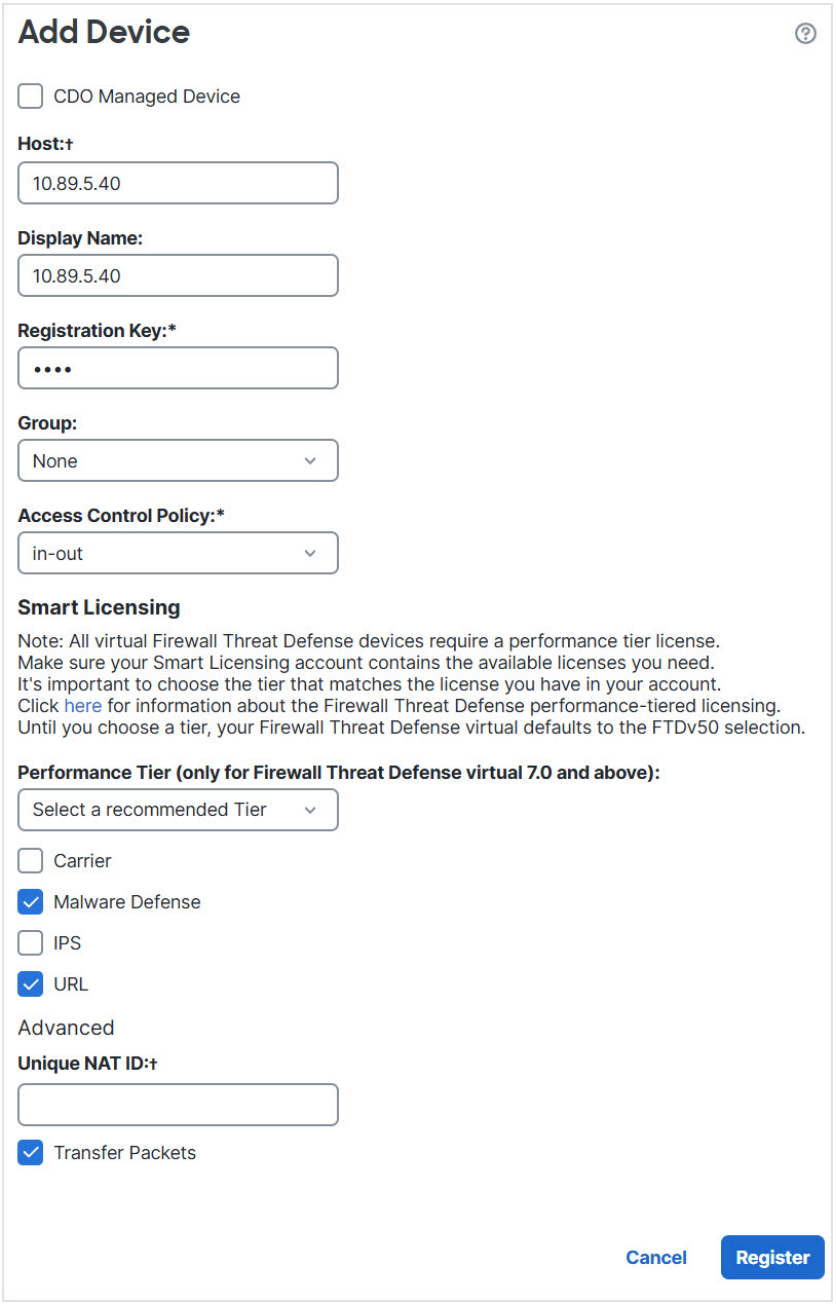
始める前に

- すべてのクラスタユニットは、Firewall Management Center に追加する前に、正常な形式のクラスタ内に存在している必要があります。また、どのユニットが制御ユニットかを確認することも必要です。Firewall Threat Defense **show cluster info** コマンドを使用します。

手順

ステップ 1 Firewall Management Center で、**[デバイス（Devices）]** > **[デバイス管理（Device Management）]** を選択してから、**[追加（Add）]** > **[デバイスの追加（Add Device）]** を選択し、制御ユニットの管理 IP アドレスを使用して制御ユニットを追加します。

図 *19* : デバイスの追加



a) [ホスト（Host）] フィールドに、制御ユニットの IP アドレスまたはホスト名を入力します。

最適なパフォーマンスを得るため、制御ユニットの追加を推奨しますが、クラスタの任意のユニットを追加できます。

デバイスのセットアップ時に NAT ID を使用した場合は、このフィールドを入力する必要がない可能性があります。詳細については、「NAT 環境」を参照してください。

b) [表示名（Display Name）] フィールドに、Firewall Management Center での制御ユニットの表示名を入力します。

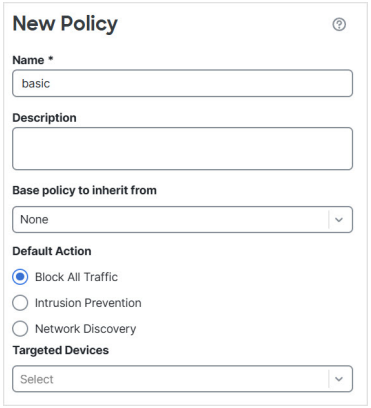
この表示名はクラスタ用ではありません。追加する制御ユニット専用です。後で、他のクラスタメンバーの名前やクラスタ表示名を変更できます。

c) [登録キー（Registration Key）] フィールドに、デバイスの設定時に使用したものと同じ登録キーを入力します。登録キーは、1 回限り使用可能な共有シークレットです。

d) （任意） デバイスをデバイス**グループ**に追加します。

e) 登録後すぐに、デバイスに展開する最初の [アクセス コントロール ポリシー（Access Control Policy）] を選択するか、新しいポリシーを作成します。

新しいポリシーを作成する場合は、基本ポリシーのみを作成します。必要に応じて、後でポリシーをカスタマイズできます。



f) デバイスに適用するライセンスを選択します。

g) デバイスの設定時に、NAT ID を使用した場合、[詳細（Advanced）] セクションを展開し、[一意の NAT ID（Unique NAT ID）] フィールドに同じ NAT ID を入力します。

h) [パケットの転送（Transfer Packets）] チェックボックスをオンにし、デバイスで Firewall Management Center にパケットを転送することを許可します。

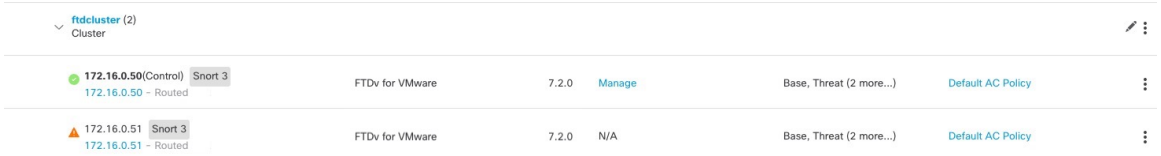
このオプションは、デフォルトで有効です。このオプションを有効にして IPS や Snort などのイベントがトリガーされた場合は、デバイスが検査用としてイベントメタデータ情報とパケットデータを Firewall Management Center に送信します。このオプションを無効にした場合は、イベント情報だけが Firewall Management Center に送信され、パケットデータは送信されません。

i) [登録（Register）] をクリックします。

Firewall Management Center は、制御ユニットを識別して登録した後に、すべてのデータユニットを登録します。制御ユニットが正常に登録されていない場合、クラスタは追加されません。クラスタが稼働状態になかった場合や、接続問題などが原因で、登録エラーが発生する場合があります。こうした状況では、クラスタユニットを再度追加することをお勧めします。

[デバイス（Devices）] > [デバイス管理（Device Management）] ページにクラスタ名が表示されます。クラスタを展開して、クラスタユニットを表示します。

図 *20* : クラスタの管理

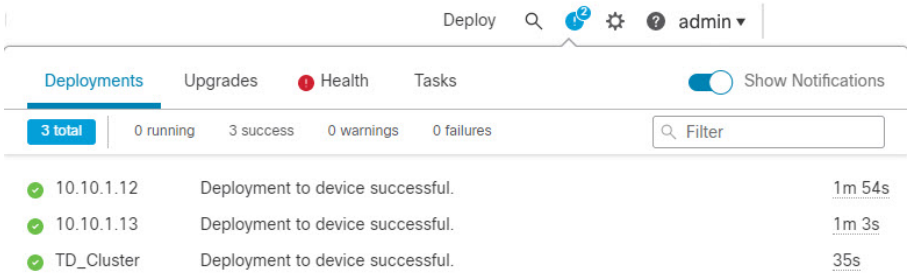


現在登録されているユニットには、ロード アイコンが表示されます。

図 *21* : ノードの登録

（注）
GCP は、クラスタノードの検出中にパブリック IP アドレスを持つノードを優先します。Firewall Threat Defense Virtual クラスタがプライベート IP アドレスを使用して Management Center Virtual に登録されるようにするには、最初に Firewall Threat Defense Virtual クラスタノードでパブリック IP アドレスを無効にする必要があります。これにより、GCP ノードの検出が Management Center Virtual への登録ノードのプライベート IP アドレスを使用して続行されます。

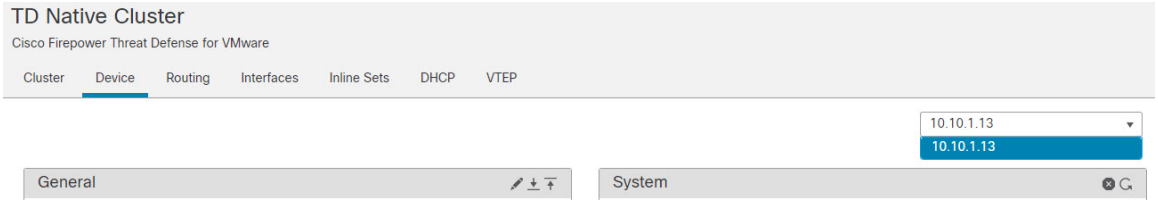
クラスタユニットの登録をモニターするには、[通知（Notifications）] アイコンをクリックし、[タスク（Tasks）] を選択します。Firewall Management Center は、ユニットの登録ごとにクラスタ登録タスクを更新します。いずれかのユニットの登録に失敗した場合には、クラスタノードの照合 （110 ページ） を参照してください。



ステップ 2 クラスタの **Edit** (✎) をクリックして、デバイス固有の設定を指定します。

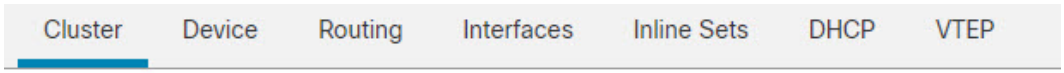
ほとんどの設定は、クラスタ内のノードではなく、クラスタ全体に適用できます。たとえば、ノードごとに表示名を変更できますが、インターフェイスはクラスタ全体についてのみ設定できます。

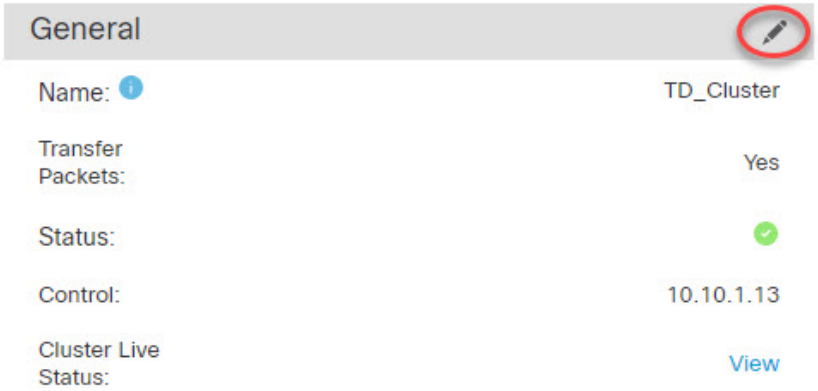
ステップ 3 **[デバイス（Devices）]** > **[デバイス管理（Device Management）]** > **[クラスタ（Cluster）]** 画面に、[全般（General）]、[ライセンス（License）]、[システム（System）]、および [ヘルス（Health）] の設定が表示されます。



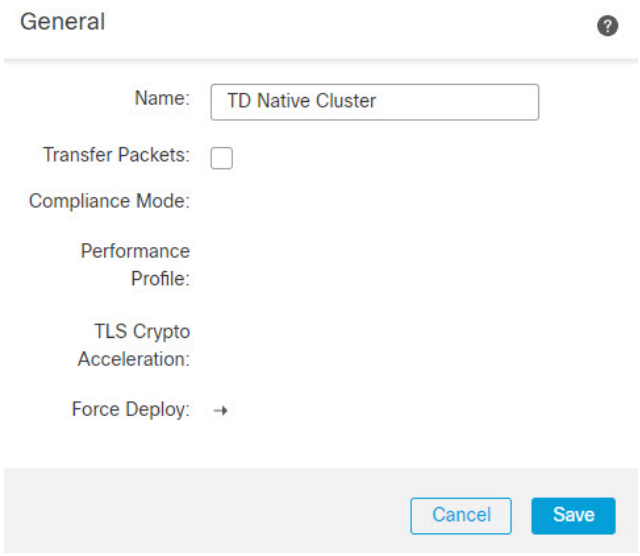
次のクラスタ固有の項目を参照してください。

- [全般（General）] > [名前（Name）]：**Edit** (✎) をクリックして、クラスタの表示名を変更します。

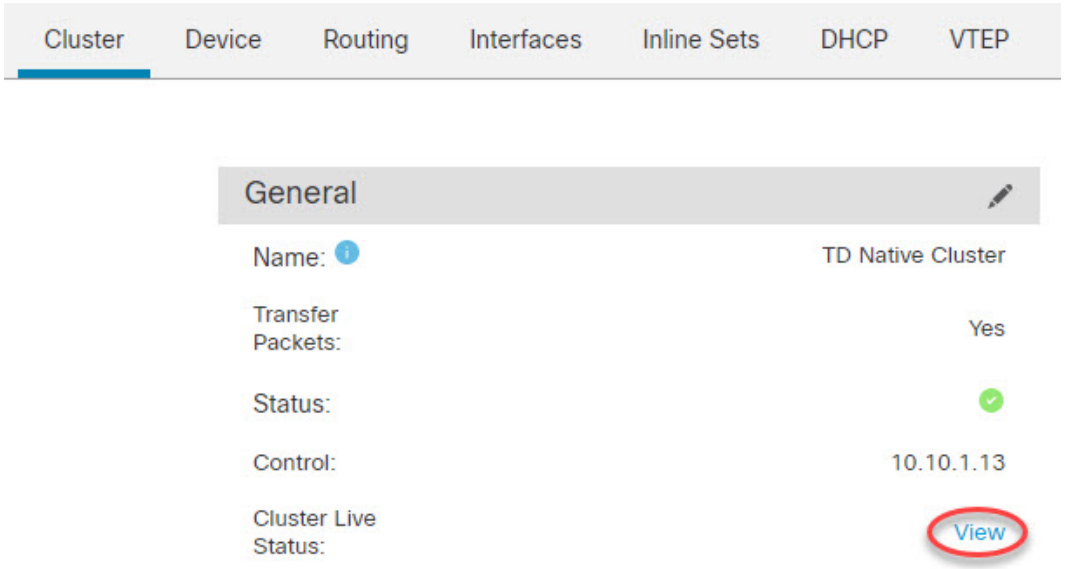




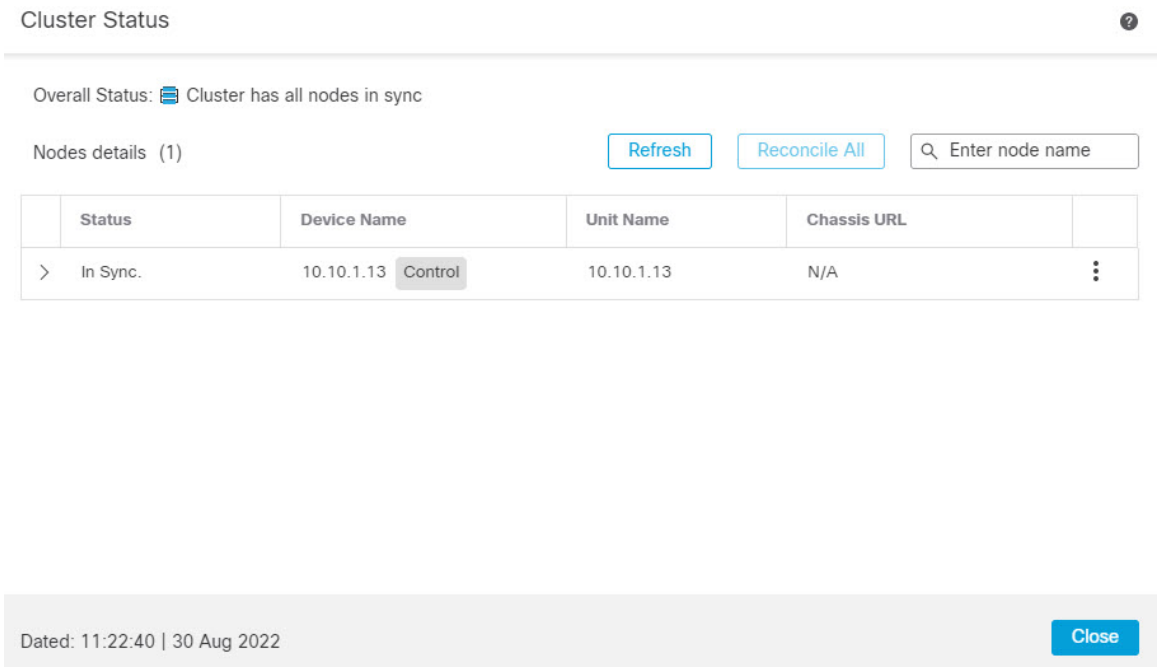
その後に、[名前（Name）] フィールドを設定します。



- [全般（General）] > [クラスタステータスの表示（View cluster status）] : [クラスタステータスの表示（View cluster status）] リンクをクリックして [クラスタステータス（Cluster Status）] ダイアログボックスを開きます。

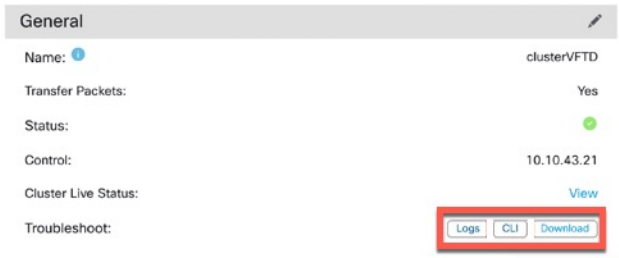


[クラスタステータス（Cluster Status）] ダイアログボックスで、[照合（Reconcile）] をクリックしてデータユニットの登録を再試行することもできます。ノードからクラスタ制御リンクに ping を実行することもできます。クラスター制御リンクへの ping の実行 （120 ページ）を参照してください。



- [全般（General）]>[トラブルシュート（Troubleshoot）]：トラブルシューティングログを生成およびダウンロードしたり、クラスタ CLI を表示したりできます。クラスタのトラブルシューティング （119 ページ）を参照してください。

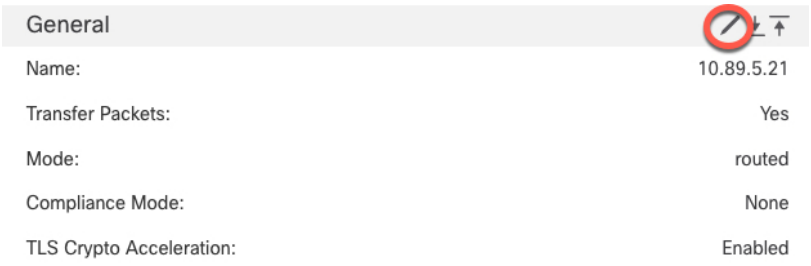
図 *22:* トラブルシューティング



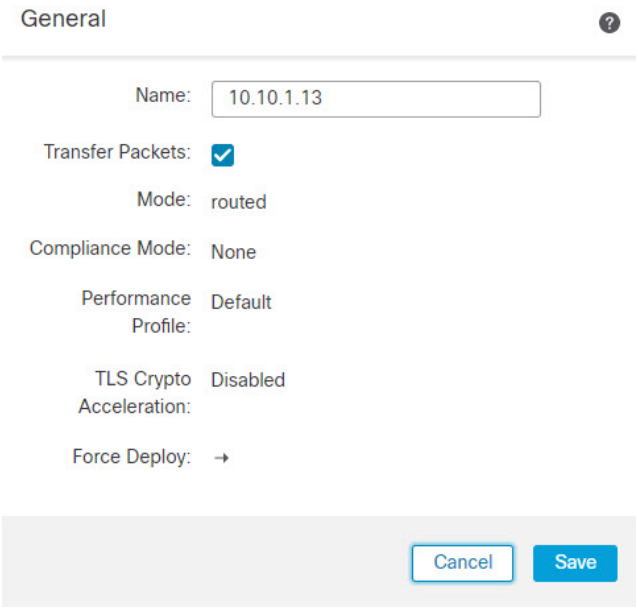
- [ライセンス（License）]：**Edit** (🔗) をクリックして、ライセンス付与資格を設定します。

ステップ 4 **[デバイス（Devices）]>[デバイス管理（Device Management）]>[デバイス（Devices）]** の右上のドロップダウンメニューで、クラスタ内の各メンバーを選択し、次の設定を指定することができます。

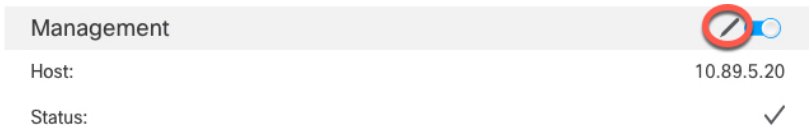
- [全般（General）]>[名前（Name）]：**Edit** (🔗) をクリックして、クラスタメンバーの表示名を変更します。



その後に、[名前（Name）] フィールドを設定します。



- [管理（Management）] > [ホスト（Host）] : デバイス設定で管理 IP アドレスを変更する場合、Firewall Management Center で新しいアドレスを一致させてネットワーク上のデバイスに到達できるようにし、[管理（Management）] 領域で [ホスト（Host）] アドレスを編集します。



クラスタのヘルスモニターの設定

[クラスタ（Cluster）] ページの [クラスタヘルスモニターの設定（Cluster Health Monitor Settings）] セクションには、次の表で説明されている設定が表示されます。

図 *23 :* クラスタのヘルスモニターの設定

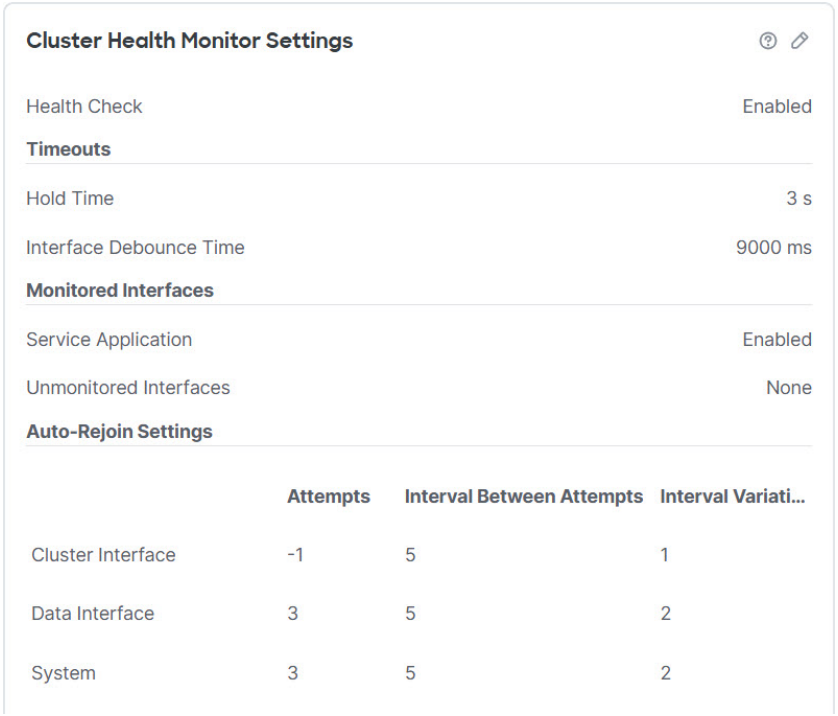


表 *6 : [*クラスタヘルスモニターの設定（***Cluster Health Monitor Settings***）*]* セクションテーブルのフィールド

フィールド	説明
タイムアウト（Timeouts）	
保留時間（Hold Time）	指定できる範囲は 0.3 ～ 45 秒です。デフォルトは 3 秒です。ノードの状態を確認するため、クラスタノードはクラスタ制御リンクで他のノードにハートビートメッセージを送信します。ノードが保留時間内にピアノードからハートビートメッセージを受信しない場合、そのピアノードは応答不能またはデッド状態と見なされます。
インターフェイスのデバウンス時間（Interface Debounce Time）	指定できる範囲は 300 ～ 9000 ミリ秒です。デフォルトは 500 ms です。インターフェイスのデバウンス時間は、インターフェイスで障害が発生していると見なされ、クラスタからノードが削除されるまでの時間です。

フィールド	説明
Monitored Interfaces（モニタリング対象インターフェイス）	インターフェイスのヘルス チェックはリンク障害をモニターします。特定の論理インターフェイスのすべての物理ポートが、特定のノード上では障害が発生したが、別のノード上の同じ論理インターフェイスでアクティブポートがある場合、そのノードはクラスタから削除されます。ノードがメンバーをクラスタから削除するまでの時間は、インターフェイスのタイプと、そのノードが確立済みノードであるか、またはクラスタに参加しようとしているかによって異なります。
サービスアプリケーション（Service Application）	Snort プロセスおよび disk-full プロセスが監視されているかどうかを示します。
モニタリング対象外のインターフェイス（Unmonitored Interfaces）	モニタリング対象外のインターフェイスを表示します。
自動再結合の設定（Auto-Rejoin Settings）	
クラスタインターフェイス（Cluster Interface）	クラスタ制御リンクに障害が発生した後に自動再結合の設定を表示します。
試行（*Attempts*）	指定できる範囲は -1 ～ 65535 です。デフォルトは -1（無制限）です。再結合の試行回数を設定します。
試行の間隔（*Interval Between Attempts*）	指定できる範囲は 2 ～ 60 です。デフォルトは 5 分です。再結合試行の間隔を分単位で定義します。
間隔のバリエーション（*Interval Variation*）	指定できる範囲は 1 ～ 3 です。デフォルトは間隔の 1 倍です。試行ごとに間隔を長くするかどうかを定義します。
データインターフェイス（Data Interfaces）	データインターフェイスに障害が発生した後に自動再結合の設定を表示します。
試行（*Attempts*）	指定できる範囲は -1 ～ 65535 です。デフォルトは 3 です。再結合の試行回数を設定します。
試行の間隔（*Interval Between Attempts*）	指定できる範囲は 2 ～ 60 です。デフォルトは 5 分です。再結合試行の間隔を分単位で定義します。
間隔のバリエーション（*Interval Variation*）	指定できる範囲は 1 ～ 3 です。デフォルトは間隔の 2 倍です。試行ごとに間隔を長くするかどうかを定義します。
システム（System）	内部エラーが発生した後に自動再結合の設定を表示します。内部エラーには、アプリケーション同期のタイムアウト、一貫性のないアプリケーションステータスなどがあります。

フィールド	説明
試行（*Attempts*）	指定できる範囲は -1 ～ 65535 です。デフォルトは 3 です。再結合の試行回数を設定します。
試行の間隔（*Interval Between Attempts*）	指定できる範囲は 2 ～ 60 です。デフォルトは 5 分です。再結合試行の間隔を分単位で定義します。
間隔のバリエーション（*Interval Variation*）	指定できる範囲は 1 ～ 3 です。デフォルトは間隔の 2 倍です。試行ごとに間隔を長くするかどうかを定義します。

（注） システムのヘルスチェックを無効にすると、システムのヘルスチェックが無効化されている場合に適用されないフィールドは表示されません。

これらの設定は、このセクションから変更できます。

任意のポートチャネル ID、単一の物理インターフェイス ID、Snort プロセス、および disk-full プロセスを監視できます。ヘルス モニタリングは VLAN サブインターフェイス、または VNI やBVI などの仮想インターフェイスでは実行されません。クラスタ制御リンクのモニタリングは設定できません。このリンクは常にモニタされています。

手順

ステップ 1 **Devices** > **Device Management** を選択します。

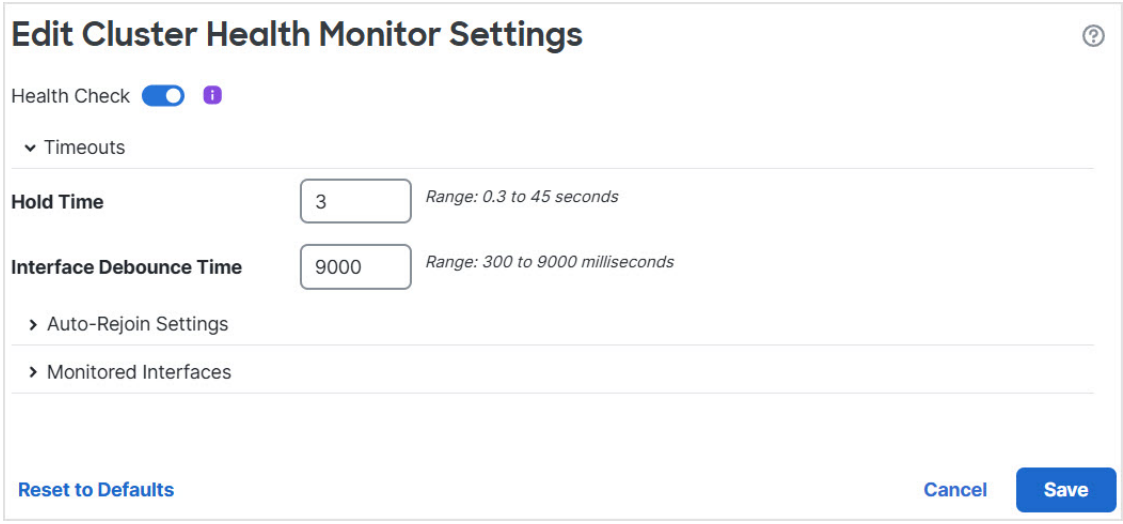
ステップ 2 変更するクラスタの横にある **Edit** (✎) をクリックします。

ステップ 3 [クラスタ（Cluster）] をクリックします。

ステップ 4 [クラスタのヘルスモニターの設定（Cluster Health Monitor Settings）] セクションで、**Edit** (✎) をクリックします。

ステップ 5 [ヘルスチェック（Health Check）] スライダをクリックして、システムのヘルスチェックを無効にします。

図 *24 :* システムヘルスチェックの無効化



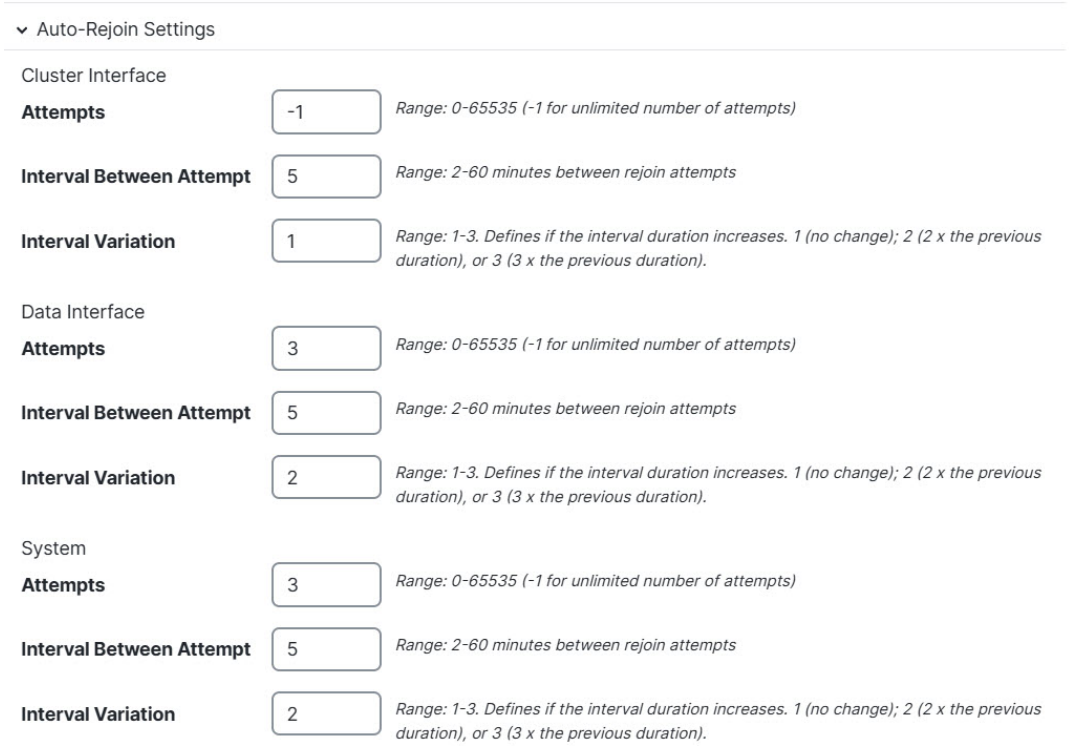
何らかのトポロジ変更（たとえばデータインターフェイスの追加/削除、ノードやスイッチのインターフェイスの有効化/無効化、VSS や vPC（または VNet）を形成するスイッチの追加）を行うときには、システムのヘルスチェック機能を無効にし、無効化したインターフェイスのモニタリングも無効にしてください。トポロジの変更が完了して、設定の変更がすべてのノードに同期されたら、システムのヘルスチェック機能を再度有効にてインターフェイスをモニタリングできます。

ステップ 6 ホールド時間とインターフェイスのデバウンス時間を設定します。

- [ホールド時間（Hold Time）]：ノードのハートビート ステータス メッセージの時間間隔を指定します。指定できる範囲は 3 ～ 45 秒で、デフォルトは 3 秒です。
- [インターフェイスのデバウンス時間（Interface Debounce Time）]：デバウンス時間は 300 ～ 9000 ms の範囲で値を設定します。デフォルトは 500 ms です。値を小さくすると、インターフェイスの障害をより迅速に検出できます。デバウンス時間を短くすると、誤検出の可能性が高くなることに注意してください。インターフェイスのステータス更新が発生すると、インターフェイス障害としてマーク付けされるまで、ノードは指定されたミリ秒数待機します。その後、ノードはクラスタから削除されます。EtherChannel がダウン状態からアップ状態に移行する場合（スイッチがリロードされた、スイッチで EtherChannel が有効になったなど）、デバウンス時間がより長くなり、ポートのバンドルにおいて別のクラスタノードの方が高速なため、クラスタノードでインターフェイスの障害が表示されることを妨げることがあります。

ステップ 7 ヘルス チェック失敗後の自動再結合クラスタ設定をカスタマイズします。

図 *25 :* 自動再結合の設定



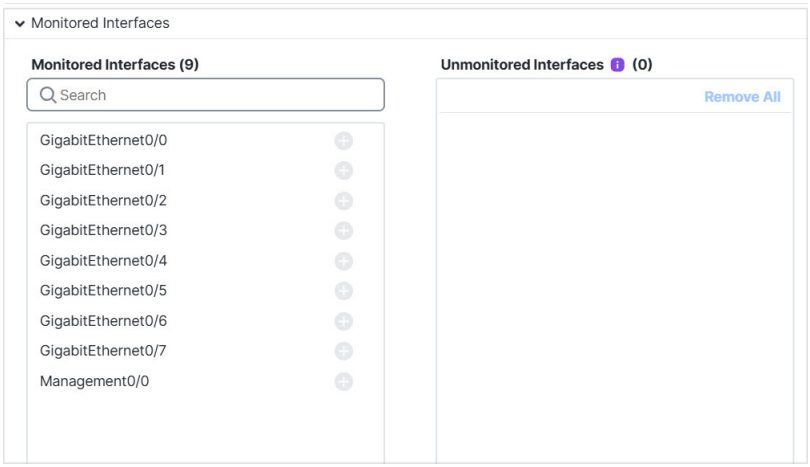
[クラスタインターフェイス（Cluster Interface）]、[データインターフェイス（Data Interface）]、および[システム（System）]に次の値を設定します（内部エラーには、アプリケーションの同期タイムアウト、一貫性のないアプリケーションステータスなどがあります）。

- [試行数（Attempts）]：再結合の試行回数を 0 ～ 65535 の範囲の値に設定します。**0** は自動再結合を無効化します。[クラスタインターフェイス（Cluster Interface）] のデフォルト値は -1（無制限）です。[データインターフェイス（Data Interface）] と [システム（System）] のデフォルト値は 3 です。
- [試行の間隔（Interval Between Attempts）]：再結合試行の間隔を 2 ～ 60 の分単位で定義します。デフォルト値は 5 分です。クラスタへの再参加をノードが試行する最大合計時間は、最後の障害発生時から 14400 分（10 日）に制限されます。
- [間隔のバリエーション（Interval Variation）]：間隔を増加させるかどうかを定義します。1 ～ 3 の範囲で値を設定します（**1**：変更なし、**2**：直前の間隔の 2 倍、**3**：直前の間隔の 3 倍）。たとえば、間隔を 5 分に設定し、変分を 2 に設定した場合は、最初の試行が 5 分後、2 回目の試行が 10 分後（2 x 5）、3 階目の試行が 20 分後（2 x 10）となります。デフォルト値は、[クラスタインターフェイス（Cluster Interface）] の場合は **1**、[データインターフェイス（Data Interface）] および [システム（System）] の場合は **2** です。

ステップ 8 [モニタリング対象のインターフェイス（Monitored Interfaces）] または [モニタリング対象外のインターフェイス（Unmonitored Interfaces）] ウィンドウでインターフェイスを移動して、モニタリング対象のインターフェイスを設定します。[サービスアプリケーションのモニタリング

を有効にする（Enable Service Application Monitoring）] をオンまたはオフにして、Snort プロセスと disk-full プロセスのモニタリングを有効または無効にすることもできます。

図 *26*：モニタリング対象インターフェイスの設定



インターフェイスのヘルスチェックはリンク障害をモニターします。特定の論理インターフェイスのすべての物理ポートが、特定のノード上では障害が発生したが、別のノード上の同じ論理インターフェイスでアクティブポートがある場合、そのノードはクラスタから削除されます。ノードがメンバーをクラスタから削除するまでの時間は、インターフェイスのタイプと、そのノードが確立済みノードであるか、またはクラスタに参加しようとしているかによって異なります。デフォルトでは、ヘルスチェックはすべてのインターフェイス、および Snort プロセスと disk-full プロセスで有効になっています。

必須以外のインターフェイスのヘルス モニタリングを無効にできます。

何らかのトポロジ変更（たとえばデータインターフェイスの追加/削除、ノードやスイッチのインターフェイスの有効化/無効化、VSS や vPC（または VNet）を形成するスイッチの追加）を行うときには、システムのヘルスチェック機能を無効にし、無効化したインターフェイスのモニタリングも無効にしてください。トポロジの変更が完了して、設定の変更がすべてのノードに同期されたら、システムのヘルスチェック機能を再度有効にてインターフェイスをモニタリングできます。

ステップ 9 [保存（Save）] をクリックします。

ステップ 10 設定変更を展開します設定変更の展開を参照してください。

クラスタノードの管理

クラスタリングを無効にする

ノードの削除に備えて、またはメンテナンスのために一時的にノードを非アクティブ化する場合があります。この手順は、ノードを一時的に非アクティブ化するためのものです。ノードは

引き続き Firewall Management Center のデバイスリストに表示されます。ノードが非アクティブになると、すべてのデータインターフェイスがシャットダウンされます。

（注） クラスタリングを無効にせずにノードの電源を切らないでください。

手順

ステップ 1 無効にするユニットに対して、[デバイス（Devices）]>[デバイス管理（Device Management）] の順に選択して **More** (⋮) をクリックし、[ノードのクラスタリングを無効にする（Disable Node Clustering）] を選択します。

ステップ 2 ノードのクラスタリングを無効にすることを確認します。

ノードは、[デバイス（Devices）]>[デバイス管理（Device Management）] リストの名前の横に [（無効（Disabled））] と表示されます。

ステップ 3 クラスタリングを再び有効にするには、クラスタへの再参加 （110ページ）を参照してください。

クラスタへの再参加

（たとえば、インターフェイスで障害が発生したために）ノードがクラスタから削除された場合、または手動でクラスタリングを無効にした場合は、クラスタに手動で再参加する必要があります。クラスタへの再参加を試行する前に、障害が解決されていることを確認します。ノードをクラスタから削除できる理由の詳細については、「クラスタへの再参加 （129ページ）」を参照してください。

手順

ステップ 1 再度有効にするユニットに対して、[デバイス（Devices）]>[デバイス管理（Device Management）] の順に選択して **More** (⋮) をクリックし、[ノードのクラスタリングを有効にする（Enable Node Clustering）] を選択します。 >

ステップ 2 ノードのクラスタリングを有効にすることを確認します。

クラスタノードの照合

クラスタノードの登録に失敗した場合は、デバイスから Firewall Management Center に対してクラスタメンバーシップを照合できます。たとえば、Firewall Management Center が特定のプロセ

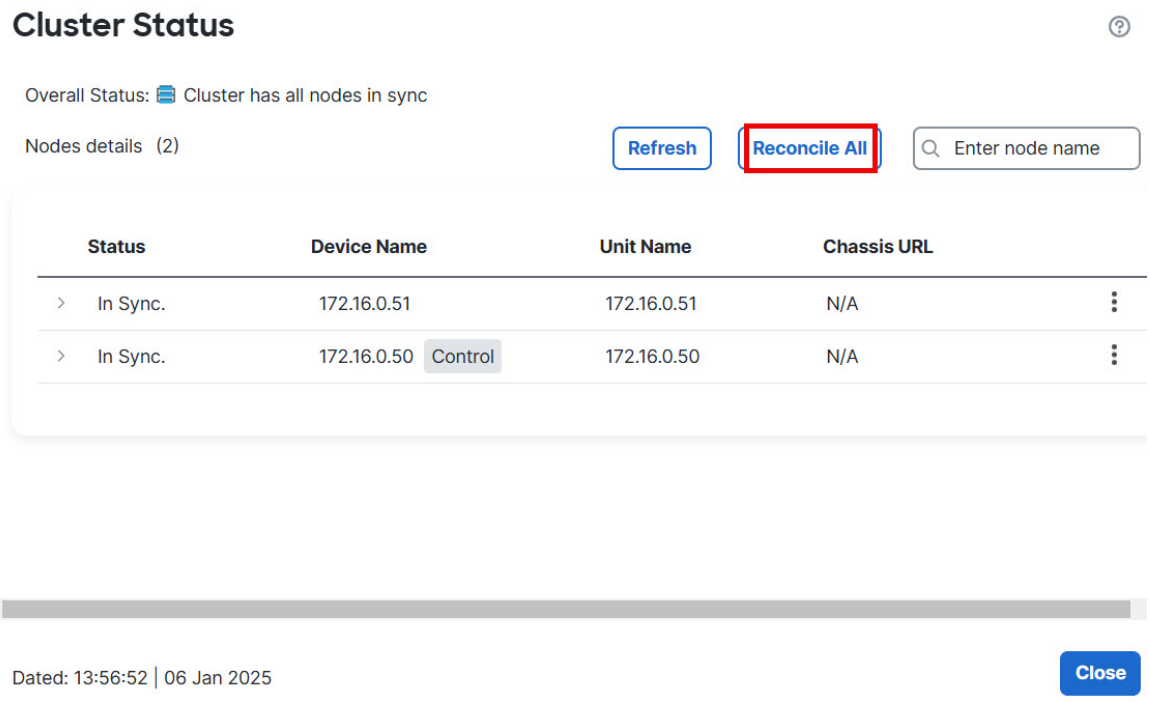
スで占領されているか、ネットワークに問題がある場合、データノードの登録に失敗することがあります。

手順

ステップ 1 クラスタの **[Devices]** > **[Device Management]** > **More** (⋮) を選択し、次に [Cluster Live Status] を選択して [Cluster Status] ダイアログボックスを開きます。

ステップ 2 [すべてを照合（Reconcile All）] をクリックします。

図 *27*：すべてを照合



クラスタ ステータスの詳細については、クラスタのモニタリング（113ページ）を参照してください。

クラスタまたはノードの登録解除と新しい Firewall Management Center への登録

Firewall Management Center からクラスタを登録解除できます。これにより、クラスタはそのまま維持されます。クラスタを新しい Firewall Management Center に追加する場合は、クラスタを登録解除することができます。

クラスタからノードを除外することなく、Firewall Management Center からノードを登録解除することもできます。ノードは Firewall Management Center に表示されていませんが、まだクラスタの一部であり、引き続きトラフィックを渡して制御ノードになることも可能です。現在動作している制御ノードを登録解除することはできません。Firewall Management Center から到達不可能になったノードは登録解除してもかまいませんが、管理接続をトラブルシューティングする間、クラスタの一部として残しておくことも可能です。

クラスタの登録解除：

- Firewall Management Center とクラスタとの間のすべての通信が切断されます。
- [デバイス管理（Device Management）] ページからクラスタが削除されます。
- クラスタのプラットフォーム設定ポリシーで、NTP を使用して Firewall Management Center から時間を受信するように設定されている場合は、クラスタがローカル時間管理に戻されます。
- 設定はそのままになるため、クラスタはトラフィックの処理を続行します。

  NAT や VPN などのポリシー、ACL、およびインターフェイス構成は維持されます。

同じまたは別の Firewall Management Center にクラスタを再登録すると、設定が削除されるため、クラスタはその時点でトラフィックの処理を停止します。クラスタ設定はそのまま維持されるため、クラスタ全体を追加できます。登録時にアクセス コントロール ポリシーを選択できますが、トラフィックを再度処理する前に、登録後に他のポリシーを再適用してから設定を展開する必要があります。

始める前に

この手順では、いずれかのノードへの CLI アクセスが必要です。

手順

ステップ 1 [デバイス（Devices）]>[デバイス管理（Device Management）]の順に選択し、クラスタかノードの **More** (⋮) をクリックして [登録解除] [削除（Delete）] を選択します。 >

ステップ 2 クラスタかノードを登録解除するよう求められたら、[はい（Yes）] をクリックします。

ステップ 3 クラスタメンバーの 1 つを新しいデバイスとして追加することにより、クラスタを新しい（または同じ）Firewall Management Center に登録できます。

クラスタノードの 1 つをデバイスとして追加するだけで、残りのクラスタノードが検出されます。

a) 1 つのクラスタノードの CLI に接続し、**configure manager add** コマンドを使用して新しい Firewall Management Center を識別します。

b) **[デバイス（Devices）]** > **[デバイス管理（Device Management）]** を選択し、[デバイスの追加（Add Device）] をクリックします。

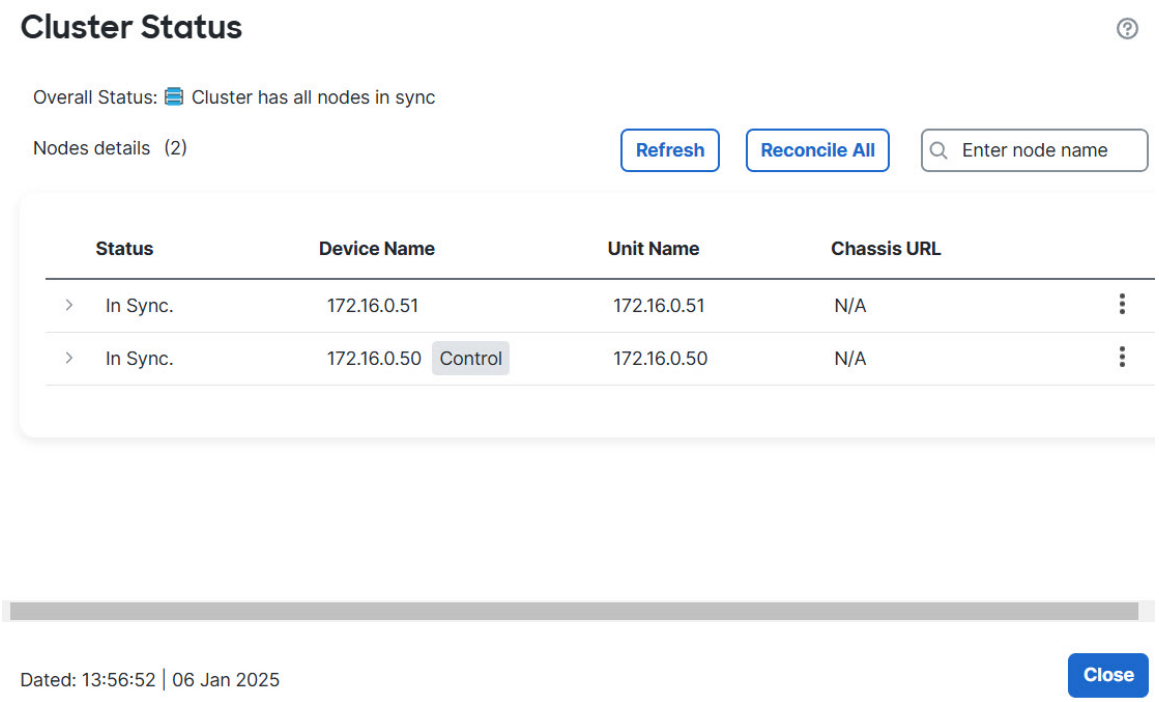
ステップ 4 削除したノードを再度追加する方法については、「クラスタノードの照合 （110ページ）」を参照してください。

クラスタのモニタリング

クラスタは、Firewall Management Center と Firewall Threat Defense の CLI でモニターできます。

- [クラスタステータス（Cluster Status）] ダイアログボックスには、[デバイス（Devices）] > [デバイス管理（Device Management）] > **More** (⋮) アイコンから、または [デバイス（Devices）] > [デバイス管理（Device Management）] > [クラスタ（Cluster）] ページ > [全般（General）] 領域 > [クラスタのライブステータス（Cluster Live Status）] リンクからアクセスできます。 > > >

図 *28 :* クラスタのステータス



制御ノードには、そのロールを示すグラフィックインジケータがあります。

クラスタメンバーステータスには、次の状態が含まれます。

- 同期中（In Sync） ：ノードは Firewall Management Center に登録されています。
- 登録の保留中（Pending Registration） ：ノードはクラスタの一部ですが、まだ Firewall Management Center に登録されていません。ノードの登録に失敗した場合は、[すべてを照合（Reconcile All）] をクリックして登録を再試行できます。

- クラスタリングが無効（Clustering is disabled）：ノードは Firewall Management Center に登録されていますが、クラスタの非アクティブなメンバーです。クラスタリング設定は、後で再有効化する予定がある場合は変更せずに維持できます。また、ノードをクラスタから削除することも可能です。
- クラスタに参加中...（Joining cluster...）：ノードがシャーシ上でクラスタに参加していますが、参加は完了していません。参加後に Firewall Management Center に登録されます。

ノードごとに [概要（Summary）] と [履歴（History）] を表示できます。

図 *29*：ノードの [概要（*Summary*）]

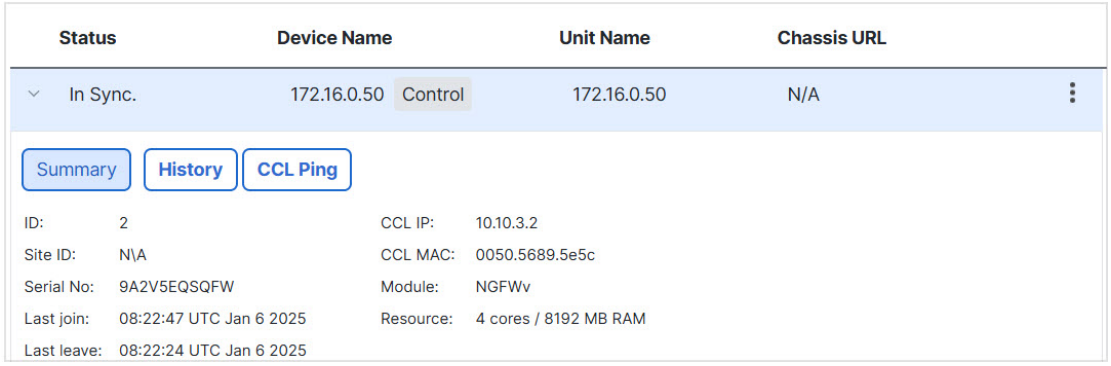
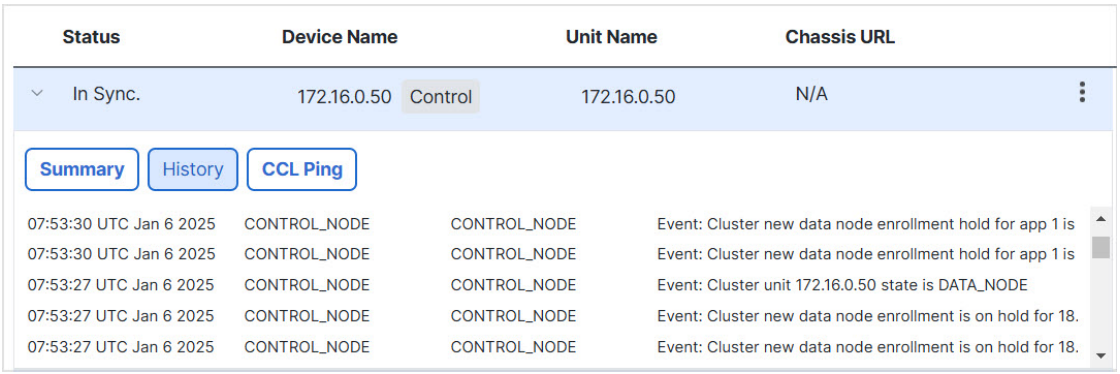


図 *30*：ノードの [履歴（*History*）]



- **System** (⚙) > [Tasks] ページ。

  [タスク（Tasks）]ページには、ノードが登録されるたびにクラスタ登録タスクの最新情報が表示されます。

- [デバイス（Devices）] > [デバイス管理（Device Management）] > cluster_name。 >

  デバイスの一覧表示ページでクラスタを展開すると、IP アドレスの横にそのロールが表示されている制御ノードを含む、すべてのメンバーノードを表示できます。登録中のノードには、ロード中のアイコンが表示されます。

- **show cluster** {**access-list** [*acl_name*] | **conn** [**count**] | **cpu** [**usage**] | **history** | **interface-mode** | **memory** | **resource usage** | **service-policy** | **traffic** | **xlate count**}

クラスタ全体の集約データまたはその他の情報を表示するには、**show cluster** コマンドを使用します。

- **show cluster info** [**auto-join** | **clients** | **conn-distribution** | **flow-mobility counters** | **goid** [*options*] | **health** | **incompatible-config** | **loadbalance** | **old-members** | **packet-distribution** | **trace** [*options*] | **transport** { **asp** | **cp**}]

  クラスタ情報を表示するには、**show cluster info** コマンドを使用します。

クラスタ ヘルス モニター ダッシュボード

クラスタのヘルスモニター

Firewall Threat Defense がクラスタの制御ノードである場合、Firewall Management Center はデバイス メトリック データ コレクタからさまざまなメトリックを定期的に収集します。クラスタの ヘルスモニターは、次のコンポーネントで構成されています。

- 概要ダッシュボード：クラスタトポロジ、クラスタ統計、およびメトリックチャートに関する情報を表示します。
  - トポロジセクションには、クラスタのライブステータス、個々の脅威防御の状態、脅威防御ノードのタイプ（制御ノードまたはデータノード）、およびデバイスの状態が表示されます。デバイスの状態は、[無効（Disabled）]（デバイスがクラスタを離れたとき）、[初期状態で追加（Added out of box）]（パブリッククラウドクラスタで Firewall Management Center に属していない追加ノード）、または [標準（Normal）]（ノードの理想的な状態）のいずれかです。
  - クラスタの統計セクションには、CPU使用率、メモリ使用率、入力レート、出力レート、アクティブな接続数、および NAT 変換数に関するクラスタの現在のメトリックが表示されます。

（注） CPU とメモリのメトリックは、データプレーンと Snort の使用量の個々の平均を示します。

  - メトリックチャート、つまり、CPU使用率、メモリ使用率、スループット、および接続数は、指定された期間におけるクラスタの統計を図表で示します。
- 負荷分散ダッシュボード：2 つのウィジェットでクラスタノード全体の負荷分散を表示します。
  - 分布ウィジェットには、クラスタノード全体の時間範囲における平均パケットおよび接続分布が表示されます。このデータは、ノードによって負荷がどのように分散されているかを示します。このウィジェットを使用すると、負荷分散の異常を簡単に特定して修正できます。

- ノード統計ウィジェットには、ノードレベルのメトリックが表形式で表示されます。クラスタノード全体の CPU 使用率、メモリ使用率、入力レート、出力レート、アクティブな接続数、および NAT 変換数に関するメトリックデータが表示されます。このテーブルビューでは、データを関連付けて、不一致を簡単に特定できます。

- メンバー パフォーマンス ダッシュボード：クラスタノードの現在のメトリックを表示します。セレクタを使用してノードをフィルタリングし、特定ノードの詳細を表示できます。メトリックデータには、CPU使用率、メモリ使用率、入力レート、出力レート、アクティブな接続数、および NAT 変換数が含まれます。
- CCL ダッシュボード：クラスタの制御リンクデータ、つまり入力レートと出力レートをグラフ形式で表示します。
- トラブルシューティングとリンク：頻繁に使用されるトラブルシューティングのトピックと手順への便利なリンクを提供します。
- 時間範囲：さまざまなクラスタ メトリック ダッシュボードやウィジェットに表示される情報を制限するための調整可能な時間枠。
- カスタムダッシュボード：クラスタ全体のメトリックとノードレベルのメトリックの両方に関するデータを表示します。ただし、ノードの選択は脅威防御メトリックにのみ適用され、ノードが属するクラスタ全体には適用されません。

クラスタ ヘルスの表示

この手順を実行するには、管理者ユーザー、メンテナンスユーザー、またはセキュリティアナリスト ユーザーである必要があります。

クラスタヘルスモニターは、クラスタとそのノードのヘルスステータスの詳細なビューを提供します。このクラスタヘルスモニターは、一連のダッシュボードでクラスタのヘルスステータスと傾向を提供します。

始める前に

- Firewall Management Center の 1 つ以上のデバイスからクラスタを作成しているかを確認します。

手順

ステップ 1 **System** (⚙) > **Health** > **Monitor** を選択します。

[モニタリング（Monitoring）] ナビゲーションウィンドウを使用して、ノード固有のヘルスモニターにアクセスします。

ステップ 2 デバイスリストで **Expand**(>) と **Collapse** (⋁) をクリックして、管理対象のクラスタデバイスのリストを展開または折りたたみます。

ステップ 3 クラスタのヘルス統計を表示するには、クラスタ名をクリックします。デフォルトでは、クラスタモニターは、いくつかの事前定義されたダッシュボードで正常性およびパフォーマンスのメトリックを報告します。メトリックダッシュボードには次のものが含まれます。

- [概要（Overview）]：他の事前定義されたダッシュボードからの主要なメトリックを表示します。ノード、CPU、メモリ、入力レート、出力レート、接続統計情報、NAT変換情報などが含まれます。
- [負荷分散（Load Distribution）]：クラスタノード間のトラフィックとパケットの分散。
- [メンバーパフォーマンス（Member Performance）]：CPU 使用率、メモリ使用率、入力スループット、出力スループット、アクティブな接続、および NAT 変換に関するノードレベルの統計情報。
- [CCL]：インターフェイスのステータスおよび集約トラフィックの統計情報。

ラベルをクリックすると、さまざまなメトリックダッシュボードに移動できます。サポートされているクラスタメトリックの包括的なリストについては、「Cisco Secure Firewall Threat Defense Health Metrics」を参照してください。

ステップ 4 右上隅のドロップダウンで、時間範囲を設定できます。最短で1時間前（デフォルト）から、最長では 2 週間前からの期間を反映できます。ドロップダウンから [Custom] を選択して、カスタムの開始日と終了日を設定します。

更新アイコンをクリックして、自動更新を 5 分に設定するか、自動更新をオフに切り替えます。

ステップ 5 選択した時間範囲について、トレンドグラフの展開オーバーレイの展開アイコンをクリックします。

展開アイコンは、選択した時間範囲内の展開数を示します。垂直の帯は、展開の開始時刻と終了時刻を示します。複数の展開の場合、複数の帯または線が表示されます。展開の詳細を表示するには、点線の上部にあるアイコンをクリックします。

ステップ 6 （ノード固有のヘルスモニターの場合）ページ上部のデバイス名の右側にあるアラート通知で、ノードの正常性アラートを確認します。

正常性アラートにポインタを合わせると、ノードの正常性の概要が表示されます。ポップアップウィンドウに、上位5つの正常性アラートの概要の一部が表示されます。ポップアップをクリックすると、正常性アラート概要の詳細ビューが開きます。

ステップ 7 （ノード固有のヘルスモニターの場合）デフォルトでは、デバイスモニターは、いくつかの事前定義されたダッシュボードで正常性およびパフォーマンスのメトリックを報告します。メトリックダッシュボードには次のものが含まれます。

- Overview：CPU、メモリ、インターフェイス、接続統計情報など、他の定義済みダッシュボードからの主要なメトリックを表示します。ディスク使用量と重要なプロセス情報も含まれます。
- CPU：CPU 使用率。プロセス別および物理コア別の CPU 使用率を含みます。
- Memory：デバイスのメモリ使用率。データプレーンと Snort のメモリ使用率を含みます。

- Interfaces：インターフェイスのステータスおよび集約トラフィック統計情報。
- Connections：接続統計（エレファントフロー、アクティブな接続数、ピーク接続数など）および NAT 変換カウント。
- [Snort]：Snort プロセスに関連する統計情報。
- [ASP ドロップ（ASP drops）]：さまざまな理由でドロップされたパケットに関連する統計情報。

ラベルをクリックすると、さまざまなメトリックダッシュボードに移動できます。サポートされているデバイスメトリックの包括的なリストについては、「Cisco Secure Firewall Threat Defense Health Metrics」を参照してください。

ステップ 8 正常性モニターの右上隅にあるプラス記号 **Add New Dashboard**(＋) をクリックして、使用可能なメトリックグループから独自の変数セットを構成し、カスタムダッシュボードを作成します。

クラスタ全体のダッシュボードの場合は、クラスタのメトリックグループを選択してから、メトリックを選択します。

クラスタメトリック

クラスタのヘルスモニターは、クラスタとそのノードに関連する統計情報と、負荷分散、パフォーマンス、および CCL トラフィックの統計データの集約結果を追跡します。

表 7: クラスタメトリック

メトリック	説明	フォーマット（Format）
CPU	クラスタノード上の CPU メトリックの平均（データプレーンと snort についてそれぞれ表示）。	パーセンテージ
メモリ	クラスタノード上のメモリメトリックの平均（データプレーンと snort についてそれぞれ表示）。	パーセンテージ
データスループット	クラスタの着信および発信データトラフィックの統計。	バイト
CCL スループット	クラスタの着信および発信 CCL トラフィックの統計。	バイト
接続	クラスタ内のアクティブな接続数。	番号
NAT 変換数	クラスタの NAT 変換数。	番号

メトリック	説明	フォーマット（Format）
分布	1 秒ごとのクラスタ内の接続分布数。	番号
パケット	クラスタ内の1秒ごとのパケット配信の件数。	番号

クラスタのトラブルシューティング

CCL Ping ツールを使用すると、クラスタ制御リンクが正しく動作していることを確認できます。 デバイスとクラスタで使用可能な次のツールを使用することもできます。

- トラブルシューティングファイル：ノードがクラスタに参加できない場合、トラブルシューティング ファイルが自動的に生成されます。また、**[デバイス（Devices）]** > **[デバイス管理（Device Management）]** > **[クラスタ（Cluster）]** > **[一般（General）]** エリアからトラブルシューティングファイルを生成してダウンロードすることもできます。トラブルシューティング ファイルの生成を参照してください。

  More (⋮) をクリックし、[トラブルシューティング ファイル（Troubleshoot Files）] を選択して、[デバイス管理（Device Management）] ページからファイルを生成することもできます。

- CLI 出力：**[デバイス（Devices）]** > **[デバイス管理（Device Management）]** > **[クラスタ（Cluster）]** > **[一般（General）]** エリアで、クラスタのトラブルシューティングに役立つ一連の定義済み CLI 出力を表示できます。クラスタに対して次のコマンドが自動的に実行されます。
  - **show running-config cluster**
  - **show cluster info**
  - **show cluster info health**
  - **show cluster info transport cp**
  - **show version**
  - **show asp drop**
  - **show counters**
  - **show arp**
  - **show int ip brief**
  - **show blocks**
  - **show cpu detailed**
  - **show interface** *ccl_interface*
  - **ping** *ccl_ip* **size** *ccl_mtu* **repeat 2**

[コマンド（Command）] フィールドに任意の **show** コマンドを入力することもできます。詳細については、CLI 出力の表示を参照してください。

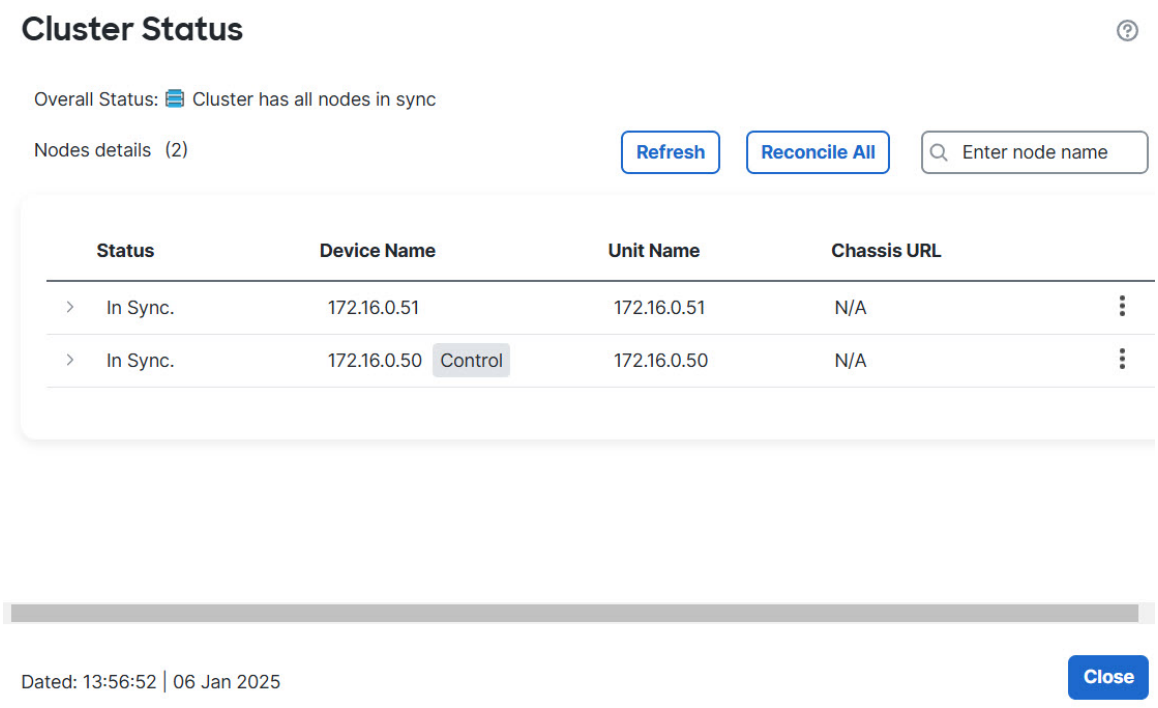
クラスター制御リンクへの ping の実行

クラスタに参加したノードは、クラスタ制御リンク MTU と一致するパケット サイズで制御ノードに ping を送信することで MTU の互換性をチェックします。ping が失敗すると、通知が生成されるため、接続スイッチの MTU 不一致を修正して再試行することができます。このツールを使用すると、クラスタ制御リンクの接続に問題がある場合に、すでにクラスタに参加しているすべてのノードに手動で ping を実行できます。

手順

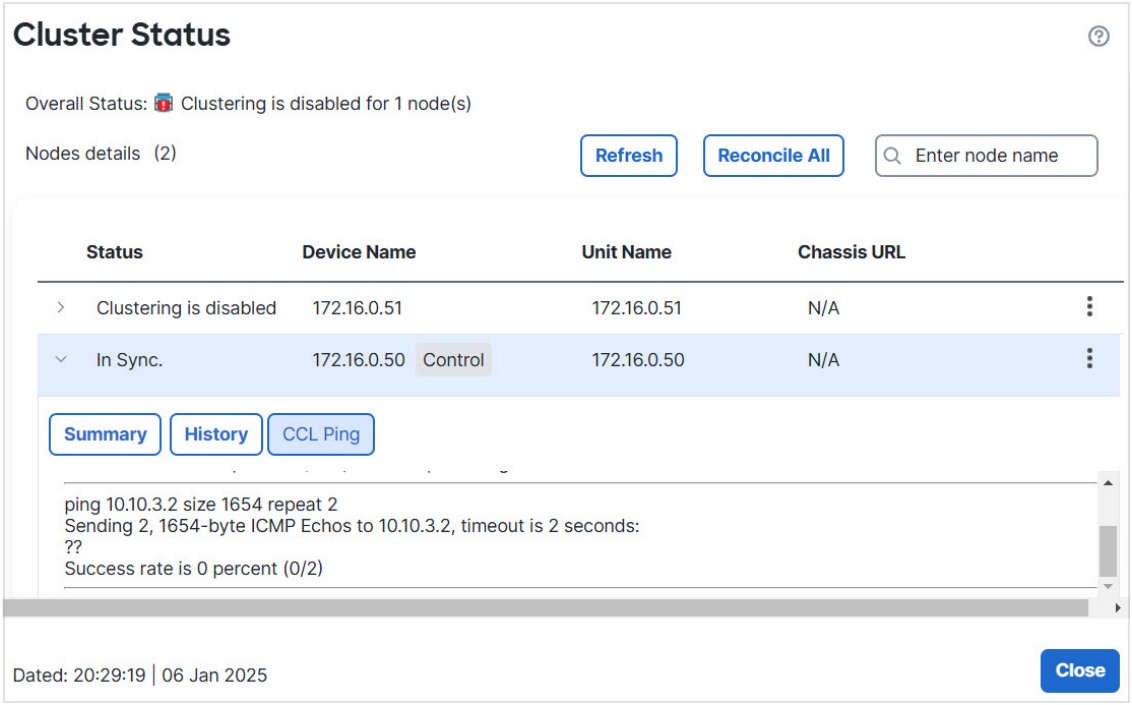
ステップ 1 **[デバイス（Devices）] > [デバイス管理（Device Management）]** の順に選択し、クラスタの横の **More** (⋮) をクリックして [クラスタのライブステータス（Cluster Live Status）] を選択します。

図 *31* : クラスタのステータス



ステップ 2 ノードの 1 つを展開し、[CCL Ping] をクリックします。

図 *32 : CCL Ping*



ノードは、最大 MTU に一致するパケットサイズを使用して、クラスタ制御リンクで他のすべてのノードに ping を送信します。

クラスタのアップグレード

Firewall Threat Defense Virtual クラスタをアップグレードするには、次の手順を実行します。

始める前に

- Before you upgrade a cluster in the public cloud, copy the target version image to your cloud image repository and update the image ID in the cluster deployment template (we actually recommend replacing the existing template with a modified copy). This ensures that after the upgrade, new instances — for example, instances launched during cluster scaling — will use the correct version. If the marketplace does not have the image you need, such as when the cluster has been patched, create a custom image from a snapshot of a standalone Firewall Threat Defense Virtual instance running the correct version, with no instance-specific (day 0) configurations.
- For Firewall Threat Defense Virtual for AWS, suspend the HealthCheck and ReplaceUnhealthy processes before autoscaled cluster upgrade. This ensures that instances are not terminated by the Auto Scaling group during the post-upgrade reboot. You can resume the suspended processes afterwards. For instructions, see the Amazon EC2 Auto Scaling user guide: Suspend and resume Amazon EC2 Auto Scaling processes.

手順

ステップ 1 ターゲット イメージ バージョンをクラウドイメージストレージにアップロードします。

ステップ 2 更新されたターゲット イメージ バージョンでクラスタのクラウド インスタンス テンプレートを更新します。

a) ターゲット イメージ バージョンを使用してインスタンステンプレートのコピーを作成します。

b) 新しく作成したテンプレートをクラスタ インスタンス グループにアタッチします。

ステップ 3 ターゲット イメージ バージョンのアップグレードパッケージを Firewall Management Center にアップロードします。

ステップ 4 アップグレードするクラスタで準備状況チェックを実行します。

ステップ 5 準備状況チェックが成功したら、アップグレードパッケージのインストールを開始します。

ステップ 6 Firewall Management Center は、クラスタノードを一度に 1 つずつアップグレードします。

ステップ 7 クラスタのアップグレードが成功すると、Firewall Management Center に通知が表示されます。

アップグレード後のインスタンスのシリアル番号と UUID に変更はありません。

クラスタリングの参考資料

このセクションには、クラスタリングの動作に関する詳細情報が含まれます。

Threat Defense の機能とクラスタリング

Firewall Threat Defense の一部の機能はクラスタリングではサポートされず、一部は制御ユニットだけでサポートされます。その他の機能については適切な使用に関する警告がある場合があります。

サポートされていない機能とクラスタリング

次の各機能は、クラスタリングが有効なときは設定できず、コマンドは拒否されます。

（注） クラスタリングでもサポートされていない FlexConfig 機能（WCCP インスペクションなど）を表示するには、ASA の一般的な操作のコンフィギュレーション ガイドを参照してください。FlexConfig では、Firewall Management Center GUI にはない多くの ASA 機能を設定できます。FlexConfig ポリシーを参照してください。

- リモート アクセス VPN（SSL VPN および IPsec VPN）

- パブリッククラウドでは、サイト間VPN（ポリシーベースおよびルートベース）はサポートされていません。
- DHCP クライアント、サーバー、およびプロキシ。DHCP リレーはサポートされています。
- 仮想トンネルインターフェイス（VTI）
- 高可用性
- 統合ルーティングおよびブリッジング
- Firewall Management Center UCAPL/CC モード

クラスタリングの中央集中型機能

The following features are only supported on the control node, and are not scaled for the cluster.

（注） Traffic for centralized features is forwarded from member nodes to the control node over the cluster control link.

If you use the rebalancing feature, traffic for centralized features may be rebalanced to non-control nodes before the traffic is classified as a centralized feature; if this occurs, the traffic is then sent back to the control node.

For centralized features, if the control node fails, all connections are dropped, and you have to re-establish the connections on the new control node.

（注） To view FlexConfig features that are also centralized with clustering, for example RADIUS inspection, see the ASA general operations configuration guide. FlexConfig lets you configure many ASA features that are not present in the Firewall Management Center GUI. See FlexConfig ポリシー.

- The following application inspections:
  - DCERPC
  - ESMTP
  - NetBIOS
  - PPTP
  - RSH
  - SQLNET
  - SUNRPC
  - TFTP
  - XDMCP

- Static route monitoring

Cisco TrustSec とクラスタリング

Only the control node learns security group tag (SGT) information. The control node then populates the SGT to data nodes, and data nodes can make a match decision for SGT based on the security policy.

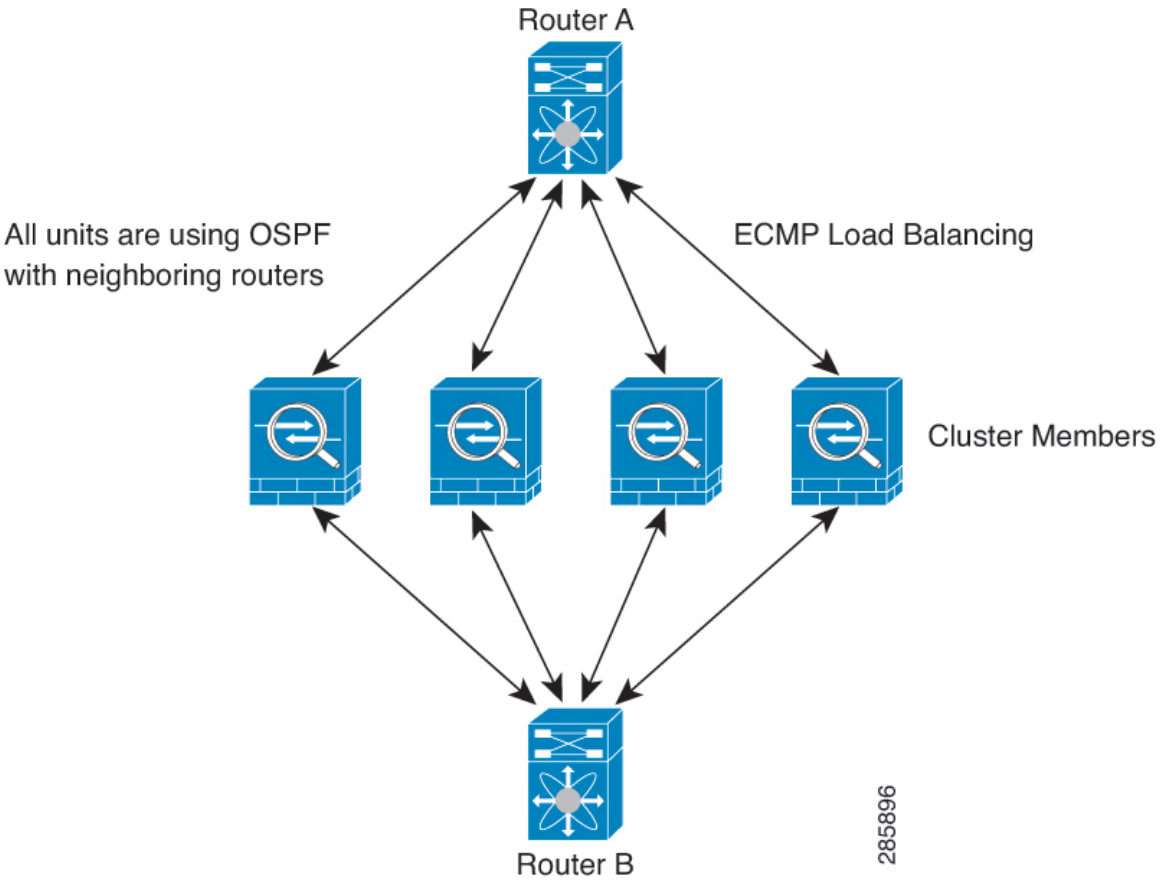
接続設定とクラスタリング

Connection limits are enforced cluster-wide. Each node has an estimate of the cluster-wide counter values based on broadcast messages. Due to efficiency considerations, the configured connection limit across the cluster might not be enforced exactly at the limit number. Each node may overestimate or underestimate the cluster-wide counter value at any given time. However, the information will get updated over time in a load-balanced cluster.

ダイナミック ルーティングおよびクラスタリング

個別インターフェイスモードでは、各ノードがスタンドアロンルータとしてルーティングプロトコルを実行します。ルートの学習は、各ノードが個別に行います。

図 *33* : 個別インターフェイス モードでのダイナミック ルーティング



上の図では、ルータ A はルータ B への等コストパスが 4 本あることを学習します。パスはそれぞれ 1 つのノードを通過します。ECMP を使用して、4 パス間でトラフィックのロード バランシングを行います。各ノードは、外部ルータと通信するときに、それぞれ異なるルータ ID を選択します。

管理者は、各ノードに異なるルータ ID が設定されるように、ルータ ID のクラスタプールを設定する必要があります。

FTP とクラスタリング

- If FTP data channel and control channel flows are owned by different cluster members, then the data channel owner will periodically send idle timeout updates to the control channel owner and update the idle timeout value. However, if the control flow owner is reloaded, and the control flow is re-hosted, the parent/child flow relationship will not longer be maintained; the control flow idle timeout will not be updated.

NAT とクラスタリング

NAT の使用については、次の制限事項を参照してください。

NAT can affect the overall throughput of the cluster. Inbound and outbound NAT packets can be sent to different Firewall Threat Defenses in the cluster, because the load balancing algorithm relies on IP addresses and ports, and NAT causes inbound and outbound packets to have different IP addresses and/or ports. When a packet arrives at the Firewall Threat Defense that is not the NAT owner, it is forwarded over the cluster control link to the owner, causing large amounts of traffic on the cluster control link. Note that the receiving node does not create a forwarding flow to the owner, because the NAT owner may not end up creating a connection for the packet depending on the results of security and policy checks.

If you still want to use NAT in clustering, then consider the following guidelines:

- プロキシ ARP なし：個別インターフェイスの場合は、マッピング アドレスについてプロキシ ARP 応答が送信されることはありません。これは、クラスタに存在しなくなった可能性のある ASA と隣接ルータとがピア関係を維持することを防ぐためです。アップストリーム ルータは、メイン クラスタ IP アドレスを指すマッピング アドレスについてはスタティック ルートまたは PBR とオブジェクト トラッキングを使用する必要があります。

- PAT with Port Block Allocation—See the following guidelines for this feature:
  - Maximum-per-host limit is not a cluster-wide limit, and is enforced on each node individually. Thus, in a 3-node cluster with the maximum-per-host limit configured as 1, if the traffic from a host is load-balanced across all 3 nodes, then it can get allocated 3 blocks with 1 in each node.
  - Port blocks created on the backup node from the backup pools are not accounted for when enforcing the maximum-per-host limit.
  - On-the-fly PAT rule modifications, where the PAT pool is modified with a completely new range of IP addresses, will result in xlate backup creation failures for the xlate backup requests that were still in transit while the new pool became effective. This behavior is not specific to the port block allocation feature, and is a transient PAT pool issue seen only in cluster deployments where the pool is distributed and traffic is load-balanced across the cluster nodes.
  - When operating in a cluster, you cannot simply change the block allocation size. The new size is effective only after you reload each device in the cluster. To avoid having to reload each

device, we recommend that you delete all block allocation rules and clear all xlates related to those rules. You can then change the block size and recreate the block allocation rules.

- NAT pool address distribution for dynamic PAT—When you configure a PAT pool, the cluster divides each IP address in the pool into port blocks. By default, each block is 512 ports, but if you configure port block allocation rules, your block setting is used instead. These blocks are distributed evenly among the nodes in the cluster, so that each node has one or more blocks for each IP address in the PAT pool. Thus, you could have as few as one IP address in a PAT pool for a cluster, if that is sufficient for the number of PAT'ed connections you expect. Port blocks cover the 1024-65535 port range, unless you configure the option to include the reserved ports, 1-1023, on the PAT pool NAT rule.
- Reusing a PAT pool in multiple rules—To use the same PAT pool in multiple rules, you must be careful about the interface selection in the rules. You must either use specific interfaces in all rules, or "any" in all rules. You cannot mix specific interfaces and "any" across the rules, or the system might not be able to match return traffic to the right node in the cluster. Using unique PAT pools per rule is the most reliable option.
- No round-robin—Round-robin for a PAT pool is not supported with clustering.
- No extended PAT—Extended PAT is not supported with clustering.
- Dynamic NAT xlates managed by the control node—The control node maintains and replicates the xlate table to data nodes. When a data node receives a connection that requires dynamic NAT, and the xlate is not in the table, it requests the xlate from the control node. The data node owns the connection.
- Stale xlates—The xlate idle time on the connection owner does not get updated. Thus, the idle time might exceed the idle timeout. An idle timer value higher than the configured timeout with a refcnt of 0 is an indication of a stale xlate.
- No static PAT for the following inspections—
  - FTP
  - RSH
  - SQLNET
  - TFTP
  - XDMCP
  - SIP
- If you have an extremely large number of NAT rules, over ten thousand, you should enable the transactional commit model using the **asp rule-engine transactional-commit nat** command in the device CLI. Otherwise, the node might not be able to join the cluster.

SIP インスペクションとクラスタリング

A control flow can be created on any node (due to load balancing); its child data flows must reside on the same node.

SNMP とクラスタリング

You should always use the Local address, and not the Main cluster IP address for SNMP polling. If the SNMP agent polls the Main cluster IP address, if a new control node is elected, the poll to the new control node will fail.

When using SNMPv3 with clustering, if you add a new cluster node after the initial cluster formation, then SNMPv3 users are not replicated to the new node. You must remove the users, and re-add them, and then redeploy your configuration to force the users to replicate to the new node.

syslog とクラスタリング

- Each node in the cluster generates its own syslog messages. You can configure logging so that each node uses either the same or a different device ID in the syslog message header field. For example, the hostname configuration is replicated and shared by all nodes in the cluster. If you configure logging to use the hostname as the device ID, syslog messages generated by all nodes look as if they come from a single node. If you configure logging to use the local-node name that is assigned in the cluster bootstrap configuration as the device ID, syslog messages look as if they come from different nodes.

VPN とクラスタリング

サイト間 VPN は、中央集中型機能です。制御ノードのみが VPN 接続をサポートします。

（注） リモート アクセス VPN は、クラスタリングではサポートされません。

VPN 機能を使用できるのは制御ノードだけであり、クラスタの高可用性機能は活用されません。制御ノードで障害が発生した場合は、すべての既存の VPN 接続が失われ、VPN ユーザにとってはサービスの中断となります。新しい制御ノードが選定されたときに、VPN接続を再確立する必要があります。

PBR または ECMP を使用するときの個別インターフェイスへの接続については、ローカル アドレスではなく、常にメイン クラスタ IP アドレスに接続する必要があります。

VPN 関連のキーと証明書は、すべてのノードに複製されます。

パフォーマンス スケーリング係数

When you combine multiple units into a cluster, you can expect the total cluster performance to be approximately 80% of the maximum combined throughput.

For example, if your model can handle approximately 10 Gbps of traffic when running alone, then for a cluster of 8 units, the maximum combined throughput will be approximately 80% of 80 Gbps (8 units x 10 Gbps): 64 Gbps.

制御ノードの選定

Nodes of the cluster communicate over the cluster control link to elect a control node as follows:

1. When you enable clustering for a node (or when it first starts up with clustering already enabled), it broadcasts an election request every 3 seconds.
2. Any other nodes with a higher priority respond to the election request; the priority is set between 1 and 100, where 1 is the highest priority.
3. If after 45 seconds, a node does not receive a response from another node with a higher priority, then it becomes the control node.

（注） If multiple nodes tie for the highest priority, the cluster node name and then the serial number is used to determine the control node.

4. If a node later joins the cluster with a higher priority, it does not automatically become the control node; the existing control node always remains as the control node unless it stops responding, at which point a new control node is elected.
5. In a "split brain" scenario when there are temporarily multiple control nodes, then the node with highest priority retains the role while the other nodes return to data node roles.

（注） You can manually force a node to become the control node. For centralized features, if you force a control node change, then all connections are dropped, and you have to re-establish the connections on the new control node.

クラスタ内のハイアベイラビリティ

クラスタリングは、ノードとインターフェイスの正常性をモニターし、ノード間で接続状態を複製することにより、ハイアベイラビリティを実現します。

ノードヘルスモニタリング

Each node periodically sends a broadcast heartbeat packet over the cluster control link. If the control node does not receive any heartbeat packets or other packets from a data node within the configurable timeout period, then the control node removes the data node from the cluster. If the data nodes do not receive packets from the control node, then a new control node is elected from the remaining nodes.

If nodes cannot reach each other over the cluster control link because of a network failure and not because a node has actually failed, then the cluster may go into a "split brain" scenario where isolated data nodes will elect their own control nodes. For example, if a router fails between two cluster locations, then the original control node at location 1 will remove the location 2 data nodes from the cluster. Meanwhile, the nodes at location 2 will elect their own control node and form their own cluster. Note that asymmetric traffic may fail in this scenario. After the cluster control link is restored, then the control node that has the higher priority will keep the control node’s role.

インターフェイス モニタリング

各ノードは、使用中のすべての指名されたハードウェアインターフェイスのリンクステータスをモニタし、ステータス変更を制御ノードに報告します。

すべての物理インターフェイスがモニタリングされます。ただし、モニタリングできるのは、名前付きインターフェイスのみです。ヘルスチェックは、インターフェイスごとに、モニターリングをオプションで無効にすることができます。

ノードのモニタ対象のインターフェイスが失敗した場合、そのノードはクラスタから削除されます。ノードは 500 ミリ秒後に削除されます。

障害後のステータス

制御ノードで障害が発生した場合、そのクラスタの他のメンバーのうち、優先順位が最高（番号が最小）のメンバーが制御ノードになります。

障害イベントに応じて、Firewall Threat Defense は自動的にクラスタへの再参加を試みます。

（注） Firewall Threat Defenseが非アクティブになり、クラスタへの自動再参加に失敗すると、すべてのデータインターフェイスがシャットダウンされ、管理インターフェイスのみがトラフィックを送受信できます。

クラスタへの再参加

クラスタ メンバがクラスタから削除された後、クラスタに再参加するための方法は、削除された理由によって異なります。

- 最初に参加するときに障害が発生したクラスタ制御リンク：クラスタ制御リンクの問題を解決した後、クラスタリングを再び有効にして、手動でクラスタに再参加する必要があります。
- クラスタに参加した後に障害が発生したクラスタ制御リンク：FTDは、無限に5分ごとに自動的に再参加を試みます。
- データ インターフェイスの障害：Firewall Threat Defense は自動的に最初は 5 分後、次に 10 分後、最終的に 20 分後に再参加を試みます。20 分後に参加できない場合、Firewall Threat Defense アプリケーションはクラスタリングを無効にします。データ インターフェイスの問題を解決した後、手動でクラスタリングを有効にする必要があります。
- ノードの障害：ノードがヘルスチェック失敗のためクラスタから削除された場合、クラスタへの再参加は失敗の原因によって異なります。たとえば、一時的な電源障害の場合は、クラスタ制御リンクが稼働している限り、ノードは再起動するとクラスタに再参加します。Firewall Threat Defense アプリケーションは 5 秒ごとにクラスタへの再参加を試みます。
- 内部エラー：内部エラーには、アプリケーション同期のタイムアウト、一貫性のないアプリケーション ステータスなどがあります。
- 障害が発生した設定の展開：FMC から新しい設定を展開し、展開が一部のクラスタメンバーでは失敗したものの、他のメンバーでは成功した場合、失敗したノードはクラスタから削除されます。クラスタリングを再度有効にして手動でクラスタに再参加する必要があります。制御ノードで展開が失敗した場合、展開はロールバックされ、メンバーは削除さ

れません。すべてのデータノードで展開が失敗した場合、展開はロールバックされ、メンバーは削除されません。

データ パス接続状態の複製

Every connection has one owner and at least one backup owner in the cluster. The backup owner does not take over the connection in the event of a failure; instead, it stores TCP/UDP state information, so that the connection can be seamlessly transferred to a new owner in case of a failure. The backup owner is usually also the director.

Some traffic requires state information above the TCP or UDP layer. See the following table for clustering support or lack of support for this kind of traffic.

表 *8 : Features Replicated Across the Cluster*

Traffic	State Support	Notes
Up time	Yes	Keeps track of the system up time.
ARP Table	Yes	—
MAC address table	Yes	—
User Identity	Yes	—
IPv6 Neighbor database	Yes	—
Dynamic routing	Yes	—
SNMP Engine ID	**No**	—

クラスタが接続を管理する方法

Connections can be load-balanced to multiple nodes of the cluster. Connection roles determine how connections are handled in both normal operation and in a high availability situation.

接続のロール

See the following roles defined for each connection:

- Owner—Usually, the node that initially receives the connection. The owner maintains the TCP state and processes packets. A connection has only one owner. If the original owner fails, then when new nodes receive packets from the connection, the director chooses a new owner from those nodes.
- Backup owner—The node that stores TCP/UDP state information received from the owner, so that the connection can be seamlessly transferred to a new owner in case of a failure. The backup owner does not take over the connection in the event of a failure. If the owner becomes unavailable, then the first node to receive packets from the connection (based on load balancing) contacts the backup owner for the relevant state information so it can become the new owner.

  As long as the director (see below) is not the same node as the owner, then the director is also the backup owner. If the owner chooses itself as the director, then a separate backup owner is chosen.

For clustering on the Firepower 9300, which can include up to 3 cluster nodes in one chassis, if the backup owner is on the same chassis as the owner, then an additional backup owner will be chosen from another chassis to protect flows from a chassis failure.

- Director—The node that handles owner lookup requests from forwarders. When the owner receives a new connection, it chooses a director based on a hash of the source/destination IP address and ports (see below for ICMP hash details), and sends a message to the director to register the new connection. If packets arrive at any node other than the owner, the node queries the director about which node is the owner so it can forward the packets. A connection has only one director. If a director fails, the owner chooses a new director.

  As long as the director is not the same node as the owner, then the director is also the backup owner (see above). If the owner chooses itself as the director, then a separate backup owner is chosen.

  ICMP/ICMPv6 hash details:

  - For Echo packets, the source port is the ICMP identifier, and the destination port is 0.
  - For Reply packets, the source port is 0, and the destination port is the ICMP identifier.
  - For other packets, both source and destination ports are 0.

- Forwarder—A node that forwards packets to the owner. If a forwarder receives a packet for a connection it does not own, it queries the director for the owner, and then establishes a flow to the owner for any other packets it receives for this connection. The director can also be a forwarder. Note that if a forwarder receives the SYN-ACK packet, it can derive the owner directly from a SYN cookie in the packet, so it does not need to query the director. (If you disable TCP sequence randomization, the SYN cookie is not used; a query to the director is required.) For short-lived flows such as DNS and ICMP, instead of querying, the forwarder immediately sends the packet to the director, which then sends them to the owner. A connection can have multiple forwarders; the most efficient throughput is achieved by a good load-balancing method where there are no forwarders and all packets of a connection are received by the owner.

（注） We do not recommend disabling TCP sequence randomization when using clustering. There is a small chance that some TCP sessions won't be established, because the SYN/ACK packet might be dropped.

- Fragment Owner—For fragmented packets, cluster nodes that receive a fragment determine a fragment owner using a hash of the fragment source IP address, destination IP address, and the packet ID. All fragments are then forwarded to the fragment owner over the cluster control link. Fragments may be load-balanced to different cluster nodes, because only the first fragment includes the 5-tuple used in the switch load balance hash. Other fragments do not contain the source and destination ports and may be load-balanced to other cluster nodes. The fragment owner temporarily reassembles the packet so it can determine the director based on a hash of the source/destination IP address and ports. If it is a new connection, the fragment owner will register to be the connection owner. If it is an existing connection, the fragment owner forwards all fragments to the provided connection owner over the cluster control link. The connection owner will then reassemble all fragments.

Port Address Translation Connections

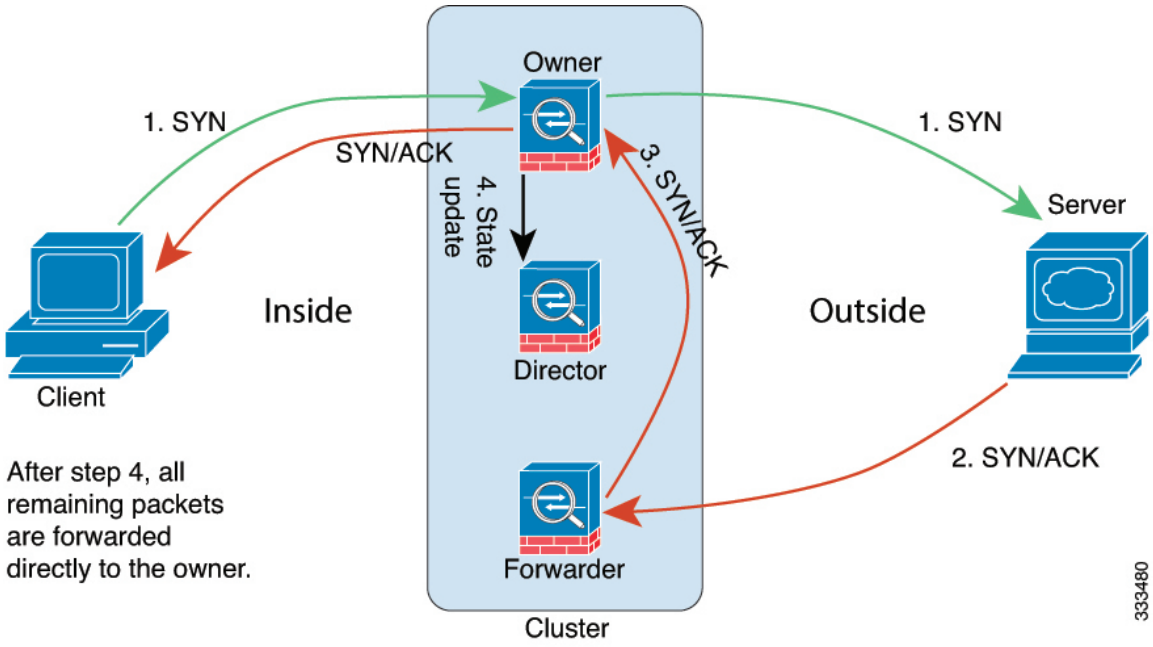
新しい接続の所有権

Traffic redirection is not supported in this release. When a new connection is directed to a node of the cluster via load balancing, that node owns both directions of the connection. All the subsequent packets for the same connection should arrive the same node. If any connection packets arrive at a different node, they will be dropped. If a reverse flow arrives at a different node, it will be dropped as well. For centralized features, if the connections do not arrive on the control node, they will be dropped.

By default, AWS GWLB uses 5-tuple to maintain flow stickiness. It is recommended to enable 2-tuple or 3-tuple stickiness on AWS GWLB to ensure the same flows are sent to the same node.

TCP のサンプルデータ フロー

The following example shows the establishment of a new connection.



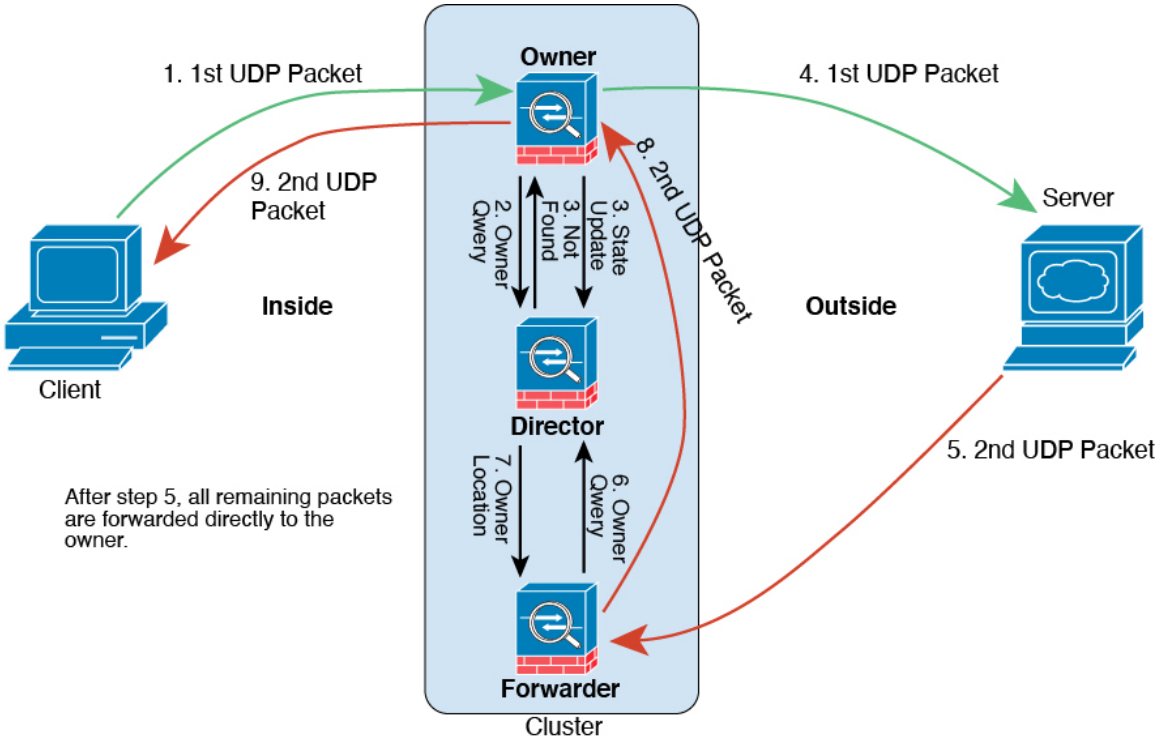
1. The SYN packet originates from the client and is delivered to one Firewall Threat Defense (based on the load balancing method), which becomes the owner. The owner creates a flow, encodes owner information into a SYN cookie, and forwards the packet to the server.
2. The SYN-ACK packet originates from the server and is delivered to a different Firewall Threat Defense (based on the load balancing method). This Firewall Threat Defense is the forwarder.
3. Because the forwarder does not own the connection, it decodes owner information from the SYN cookie, creates a forwarding flow to the owner, and forwards the SYN-ACK to the owner.
4. The owner sends a state update to the director, and forwards the SYN-ACK to the client.
5. The director receives the state update from the owner, creates a flow to the owner, and records the TCP state information as well as the owner. The director acts as the backup owner for the connection.
6. Any subsequent packets delivered to the forwarder will be forwarded to the owner.
7. If packets are delivered to any additional nodes, it will query the director for the owner and establish a flow.

8. Any state change for the flow results in a state update from the owner to the director.

ICMP および UDP のサンプルデータフロー

The following example shows the establishment of a new connection.

1. 図 *34 : ICMP and UDP Data Flow*



The first UDP packet originates from the client and is delivered to one Firewall Threat Defense (based on the load balancing method).

2. The node that received the first packet queries the director node that is chosen based on a hash of the source/destination IP address and ports.
3. The director finds no existing flow, creates a director flow and forwards the packet back to the previous node. In other words, the director has elected an owner for this flow.
4. The owner creates the flow, sends a state update to the director, and forwards the packet to the server.
5. The second UDP packet originates from the server and is delivered to the forwarder.
6. The forwarder queries the director for ownership information. For short-lived flows such as DNS, instead of querying, the forwarder immediately sends the packet to the director, which then sends it to the owner.
7. The director replies to the forwarder with ownership information.
8. The forwarder creates a forwarding flow to record owner information and forwards the packet to the owner.

9. The owner forwards the packet to the client.

パブリッククラウドの Threat Defense Virtual クラスタリングの履歴

表 *9:*

機能	Minimum Firewall Management Center	Minimum Firewall Threat Defense	詳細
MTU ping test on cluster node join	7.6.0	7.6.0	When a node joins the cluster, it checks MTU compatibility by sending a ping to the control node with a packet size matching the cluster control link MTU. If the ping fails, a notification is generated so you can fix the MTU mismatch on connecting switches and try again.
Cluster control link ping tool.	7.2.6/7.4.1	いずれか	You can check to make sure all the cluster nodes can reach each other over the cluster control link by performing a ping. One major cause for the failure of a node to join the cluster is an incorrect cluster control link configuration; for example, the cluster control link MTU may be set higher than the connecting switch MTUs. New/modified screens: **Devices** > **Device Management** > **More** > **Cluster Live Status.**
Troubleshooting file generation and download available from Device and Cluster pages.	7.4.1	7.4.1	You can generate and download troubleshooting files for each device on the Device page and also for all cluster nodes on the Cluster page. For a cluster, you can download all files as a single compressed file. You can also include cluster logs for the cluster for cluster nodes. You can alternatively trigger file generation from the **Devices** > **Device Management** > **More** > **Troubleshoot Files** menu. New/modified screens: • **Devices** > **Device Management** > **Device** > **General** • **Devices** > **Device Management** > **Cluster** > **General**
View CLI output for a device or device cluster.	7.4.1	任意	You can view a set of pre-defined CLI outputs that can help you troubleshoot the device or cluster. You can also enter any **show** command and see the output. New/modified screens: **Devices** > **Device Management** > **Cluster** > **General**

機能	Minimum Firewall Management Center	Minimum Firewall Threat Defense	詳細
クラスタのヘルスモニターの設定。	7.3.0	いずれか	クラスタのヘルスモニター設定を編集できるようになりました。 新規/変更された画面：**[デバイス（Devices）]**>**[デバイス管理（Device Management）]** > **クラスタ（Cluster）** > **[クラスタのヘルスモニターの設定（Cluster Health Monitor Settings）]** （注） 以前にFlexConfigを使用してこれらの設定を行った場合は、展開前に必ずFlexConfigの設定を削除してください。削除しなかった場合は、FlexConfig の設定によって Management Center の設定が上書きされます。
クラスタ ヘルス モニターダッシュボード。	7.3.0	いずれか	クラスタのヘルス モニター ダッシュボードでクラスタの状態を表示できるようになりました。 新規/変更された画面：**[システム（System）]** > **[正常性（Health）]** > **[モニター（Monitor）]**
Azure の Firewall Threat Defense Virtual のクラスタリング	7.3.0	7.3.0	Azure ゲートウェイロードバランサまたは外部のロードバランサについて、Azure の Firewall Threat Defense Virtual で最大 16 ノードのクラスタリングを構成できるようになりました。 新規/変更された画面： • [デバイス（Devices）] > [デバイス管理（Device Management）] > [クラスタの追加（Add Cluster）] • **[デバイス（Devices）]** > **[デバイス管理（Device Management）]** > **[詳細（More）]**メニュー • **[Devices]** > **[Device Management]** > **[Cluster]** サポートされているプラットフォーム：Azure の Firewall Threat Defense Virtual

機能	Minimum Firewall Management Center	Minimum Firewall Threat Defense	詳細
パブリッククラウドでの Firewall Threat Defense Virtual のクラスタリング（Amazon Web Services および Google Cloud Platform）。	7.2.0	7.2.0	Firewall Threat Defense Virtual はパブリッククラウド（AWS および GCP）で最大 16 ノードの個別インターフェイスのクラスタリングをサポートします。 新しい/変更された画面： • **[デバイス（Devices）]** > **[デバイス管理（Device Management）]** > **[デバイスの追加（Add Device）]** • **[デバイス（Devices）]** > **[デバイス管理（Device Management）]** > **[詳細（More）]**メニュー • **[Devices]** > **[Device Management]** > **[Cluster]** サポートされているプラットフォーム：AWS および GCP 上の Firewall Threat Defense Virtual

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。