VPC-DI の概要

この章では、Cisco Virtualized Packet Core—Distributed Instance（VPC-DI）について説明します。VPC-DI は、単一の仮想マシン（VM）を超えて境界を拡張することにより、仮想化されたクラウドアーキテクチャのスケーラビリティをサポートします。

製品の説明

この章では、StarOS VPC-DI アーキテクチャと外部デバイスとの連携動作について説明します。

VPC-DI は、1 つの仮想マシン（VM）という枠を超えて仮想化された StarOS を配布します。これにより、共有インターフェイス、共有サービスアドレス、ロードバランシング、冗長性、および管理の一元化を使用して、複数の VM を 1 つの StarOS インスタンスとして機能させることができます。

システムは、1 つの StarOS インスタンスを形成するためにグループ化された複数の VM から成る完全分散ネットワークとして動作し、各タイプのスタンバイ VM で管理およびセッション処理を実行する VM を備えています。

システムの基礎となるインフラストラクチャ

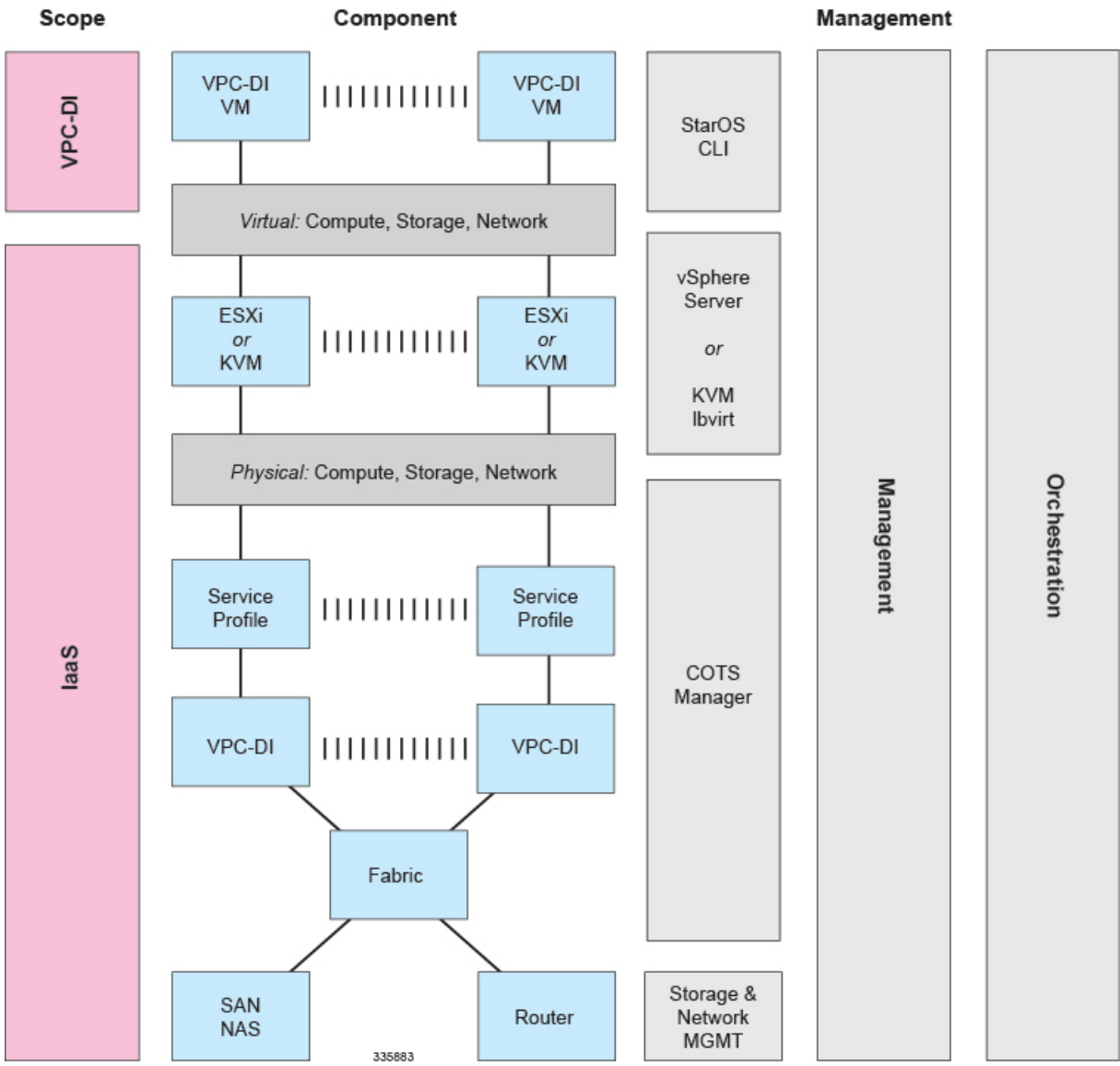
この仮想化システムは、新規または既存の Infrastructure as a Service（IaaS; サービスとしてのインフラストラクチャ）のクラウドデータセンターに展開できます。VPC-DIは、既製品（COTS）のサーバで業界標準のハイパーバイザを使用して、一連の仮想マシン（VM）で実行されます。この導入モデルでは、物理インフラストラクチャの管理をStarOSとVPC-DIの範囲外に維持することができ ます。

一般的なインスタンスは、次の NFVi（ネットワーク機能の仮想インフラストラクチャ）で実行されます。

- IaaS コンポーネント
  - COTS ブレードシャーシ、ブレード、およびファブリックインターコネクト
  - Manager ソフトウェア
  - ネットワーク接続ストレージ（NAS）またはストレージエリアネットワーク（SAN）
  - 各ブレードまたはサーバ上の VMWare ESXi または KVM ハイパーバイザ
  - VMware vSphere サーバまたは KVM OpenStack
- VM 内にインストールされた StarOS ソフトウェア
- ブレードに障害が発生した場合、単一のVMのみが影響を受けるように、各VMは別々のブレード上で実行する必要があります（アンチアフィニティ）。
- 既存の管理ソフトウェア（サービス、ロギング、統計情報など）
- オーケストレーション ソフトウェア（オプション）（オーケストレーション （19 ページ） を参照）

VPC-DI インスタンスは、StarOS の単一の管理可能なインスタンスとして機能する VM のグループです。VPC-DI は、 次の主要なコンポーネントで構成されています。

図 *1 : VPC-DI* の範囲



- 制御機能（CF）の VM （3 ページ）
- サービス機能（SF） VM （4 ページ）
- DI ネットワーク （6 ページ）

制御機能（CF）の VM

2 つの CF VM がアクティブ/スタンバイ（1:1）冗長ペアとして機能します。アクティブ CF は次の機能を果たします。

- コントローラのタスク

- ローカルコンテキスト VPNMGR
- ローカルコンテキスト（MGMT）と DI ネットワーク vNIC
- vHDD のシステム起動イメージと設定ストレージ
- vHDD のレコードストレージ
- CLI およびロギング用のアウトオブバンド（OOB） 管理（vSerial と vKVM）

サービス機能（SF） VM

SF VM は、サービスコンテキスト（ユーザ I/O ポート）を提供し、プロトコルシグナリングおよびセッション処理タスクに対応します。VPC-DI インスタンスは最小 4、最大 14 の SF VM を持つことができ、最大 12 の SF VM をアクティブにできます。

各 SF VM は、CF に指定されているとおりに 3 つのロールのいずれかを動的に実行します。

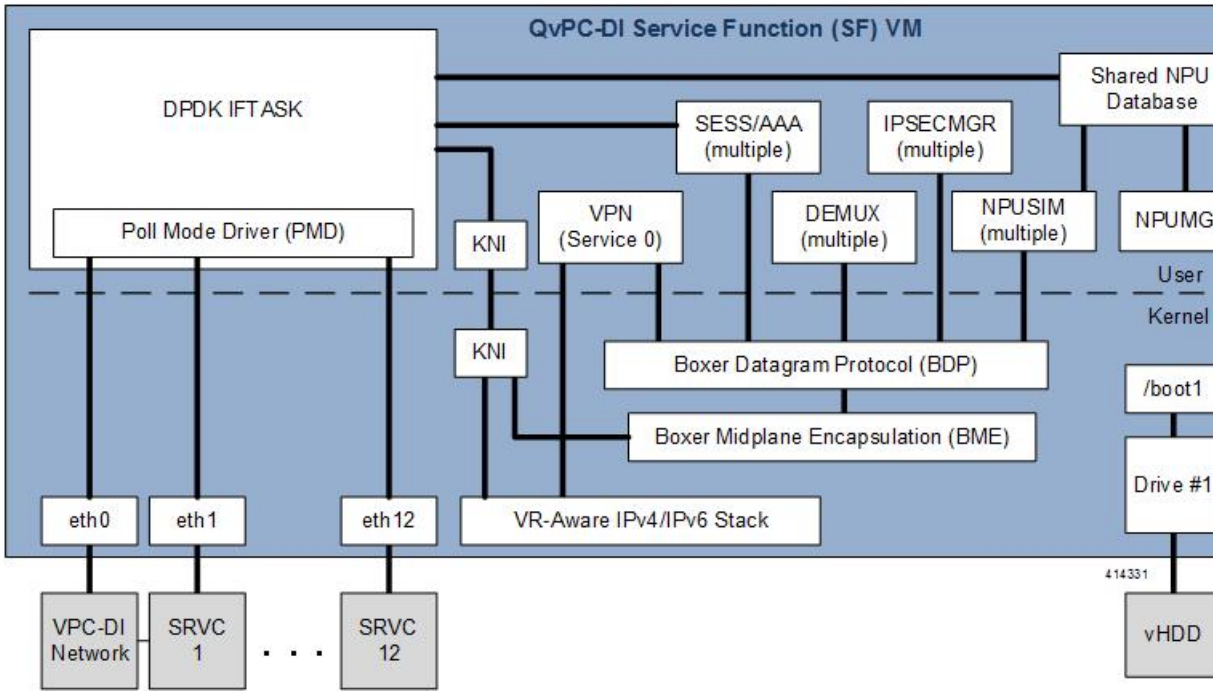
- Demux VM（フロー割り当て）
- セッション VM（トラフィック処理）
- スタンバイ VM（n+1 冗長性）

SF は次の機能を担当します。

機能	実行場所
NPUSIM fastpath/slow パス（NPU エミュレーションと CPU へのルーティング）	Demux VM、セッション VM、スタンバイ VM
Intel® Data Plane Development Kit（DPDK）に基づく IFTASK	Demux VM、セッション VM、スタンバイ VM
非ローカルコンテキスト（SRVC）の vNIC ポート	Demux VM、セッション VM、スタンバイ VM
サービスコンテキスト（最初の VM）の VPNMGR および Demux	Demux VM
セッション処理のための SESSMGR および AAAMGR（追加の VM）	セッション VM
出力転送の決定	
暗号処理	

VPC-DI インスタンスの最小設定では、4 つの SF の 2 つのアクティブ、1 つの demux 、および 1 つのスタンバイが必要です。

図 *2:* サービス機能 *VM*



DPDK 内部フォワーダ

Intel Data Plane Development Kit（DPDK）は、VPC アーキテクチャの不可欠な部分であり、システムパフォーマンスを向上させるために使用されます。DPDK 内部フォワーダ（IFTASK）は、パケットの入出力操作を担当するソフトウェアコンポーネントであり、Linux カーネルをバイパスすることによって、ユーザ空間でのパケット処理の高速パスを提供します。これはシステムの運用に必要です。CF または SF のインスタンス化時に、DPDK は CPU コアの合計数に応じて、CPU コアの特定の割合を IFTASK に割り当てます。残りの CPU コアはアプリケーションに割り当てられます。

IFTASK で使用されている CPU コアを特定し、それらの使用率を表示するには、**show npu utilization table** コマンドを次のように使用します。

```
[local]mySystem# show npu utilization table

Wednesday July 06 10:53:55 PDT 2017
          -------iftask-------
   lcore     now   5min  15min
--------  ------ ------ ------
  01/0/1     38%    53%    52%
  01/0/2     51%    55%    55%
  02/0/1     66%    72%    68%
  02/0/2     66%    63%    67%
  03/0/1     57%    55%    55%
  03/0/2     51%    47%    45%
  03/0/3     89%    89%    89%
  03/0/4     88%    88%    89%
  04/0/1     67%    59%    58%
  04/0/2     54%    40%    48%
```

```
  04/0/3     89%    89%    90%
  04/0/4     90%    89%    89%
  05/0/1     55%    55%    56%
  05/0/2     68%    45%    45%
  05/0/3     90%    89%    89%
  05/0/4     90%    89%    89%
  06/0/1     50%    58%    58%
  06/0/2     24%    24%    25%
  06/0/3     89%    90%    90%
  06/0/4     91%    90%    90%
```

IFTASK コアを使用せずに VM の CPU 使用率を表示するには、**show cpu info** コマンドを使用します。詳細については、**verbose** キーワードを使用してください。

```
[local]mySystem# show cpu info card 6
Tuesday July 05 10:39:52 PDT 2017
Card 6, CPU 0:
  Status                : Active, Kernel Running, Tasks Running
  Load Average          : 7.74, 7.62, 7.54 (9.44 max)
  Total Memory          : 49152M
  Kernel Uptime         : 4D 5H 7M
  Last Reading:
    CPU Usage           : 25.4% user, 7.8% sys, 0.0% io, 0.1% irq, 66.7% idle
    Poll CPUs           : 4 (1, 2, 3, 4)
    Processes / Tasks   : 177 processes / 35 tasks
    Network             : 164.717 kpps rx, 1025.315 mbps rx, 164.541 kpps tx, 1002.149
mbps tx
    File Usage          : 8256 open files, 4941592 available
    Memory Usage        : 21116M 43.0% used
  Maximum/Minimum:
    CPU Usage           : 32.9% user, 8.9% sys, 0.0% io, 0.4% irq, 59.1% idle
    Poll CPUs           : 4 (1, 2, 3, 4)
    Processes / Tasks   : 184 processes / 36 tasks
    Network             : 178.388 kpps rx, 1270.977 mbps rx, 178.736 kpps tx, 1168.999
mbps tx
    File Usage          : 8576 open files, 4941272 available
    Memory Usage        : 21190M 43.1% used
```

DI ネットワーク

VPC-DI インスタンス内の VM が相互に通信するためには、各インスタンスに VM をインターコネクトするプライベート L2 ネットワークが必要です。このネットワークは、IaaS/仮想化インフラストラクチャ内の VLAN を使用し、最初の vNIC として各 VM にタグなしで公開されるようにする必要があります。

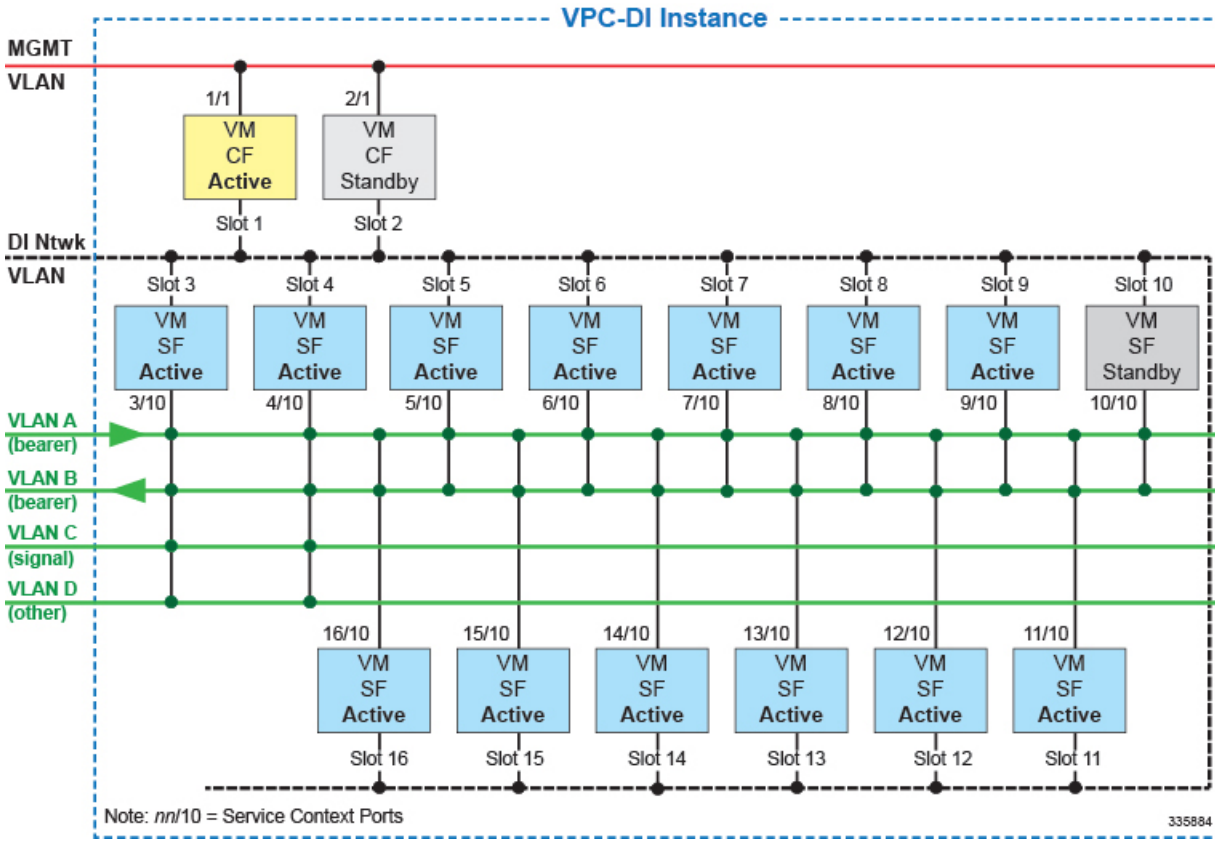
DI ネットワークは、単一の VPC-DI インスタンスを排他的に使用する必要があります。他のデバイスがこのネットワークに接続されていない可能性があります。

（注） 1 つのデータセンター内で複数のインスタンスをインスタンス化する場合は、各インスタンスに独自の DI ネットワークが必要です。

インスタンス内のすべての VM は、物理的に同じサイトに配置する必要があり、理想的には最小限の相互接続デバイスとともに同じいくつかのラックに配置します。DI ネットワークの信

頼性は、VPC-DIインスタンスの安定性を確保するために重要です。WANまたは輻輳状態のリンク全体でL2トンネリングプロトコルを使用することは推奨されません。

図 *3 : DI* ネットワーク



ネットワーク要件

DI ネットワークの信頼性とパフォーマンスは、VPC-DI の信頼性とパフォーマンスにとって重要です。DI ネットワークは、内部制御、シグナリング、およびベアラートラフィックに使用されます。ベアラートラフィックは DI ネットワークを複数回通過する可能性があるため、DI ネットワーク内のパケット損失が認識可能な VPC-DI のパケット損失全体に影響します。

OpenStack での展開の場合、DI ネットワークとサービスネットワークの両方のタイプを SR-IOV モードフラットとして設定する必要があります。VMware での展開の場合、DI ネットワークタイプは、ボンディングおよび VLAN タグ付きの VMXNET3 または PCI-PT として設定する必要があります。VMware では、サービスネットワークタイプを VMXNET3 または PCI-PT として設定する必要があります。

（注） VM を接続するインフラストラクチャは、すべての VM 間で 10 Gbps 以上であり、冗長性の設定が行われている必要があります。冗長性の設定は、次のいずれかの方法で実装できます。

- vSwitch を使用しているホスト上（Virtio/VMXNET3 インターフェイスの場合）
- Cisco UCS 仮想インターフェイスカード（VIC）などのハードウェア上
- ネットワーク インターフェイスのボンディンを使用している VPC-DI 内

IaaS/ハイパーバイザには、次が可能な DI ネットワークが備わっている必要があります。

- L2 イーサネットブリッジ/スイッチとして実行できる。
- 少なくとも 7100 バイトのジャンボフレームをサポートできる。インストールでジャンボフレームがサポートされていない場合でも、VPC-DI を使用できます。サポート対象MTUを超えるトラフィックのサポートの設定の説明に従って、起動パラメータファイルに適切なパラメータを設定する必要があります。

インフラストラクチャ/ハイパーバイザには、次が可能な DI ネットワークが備わっている必要があります。

- VID = 0 の VM から送信された優先順位 802.1p をサポートしている。
- インスタンス内のすべての VM 間で優先順位 802.1p のエンドツーエンドで受け入れる。
- すべての物理インフラストラクチャ内に冗長 L2 パスが備わっているか、またはインスタンス内のすべてのシステム VM 間で優先順位 802.1p のエンドツーエンドが備わっている。
- アクセスを制限するセキュアなネットワークが備わっている。ローカルでのみ使用できるという意味で、セキュアなネットワークである必要があります。

具体的には、DI ネットワークには次の最低限の信頼性要件が必要です。

- 完全冗長 L2 パス
- STP や LACP の停止を含めて、1.5 秒を超える停止がない（該当する場合）
- パケットの優先順位付け
- 制御パケットまたはベアラーパケットの損失を最小限に抑えるために十分なネットワーク帯域幅

DI ネットワーク内での中断や過度なパケット損失が原因で、VPC-DI インスタンスでの障害の誤検出や、想定外の動作が発生する場合があります。

各システム VM は、他の VM の到達可能性や DI ネットワークの信頼性を継続的にモニタします。

ジャンボ フレーム

DI ネットワークでは、少なくとも 7100 バイトのジャンボフレームをサポートすることを推奨します。起動時に、各 VM はネットワーク上で一連の ping コマンドを発行して、ジャンボフレームがサポートされているかどうかを確認します。ジャンボフレームがサポートされていれば、システムパフォーマンスが向上します。

インストールでジャンボフレームがサポートされていない場合でも、VPC-DI を使用できます。サポート対象 MTU を超えるトラフィックのサポートの設定の説明に従って、起動パラメータファイルに適切なパラメータを設定する必要があります 。

7100 未満の MTU が検出され、適切な起動パラメータが設定されていない場合、CF と SF は起動しません。

また、StarOS 設定で適切に設定されている場合、SF のサービスポートは、リリース 21.4 以降で最大 9,100バイトの最大 MTU、古いリリースでは 2,048 バイトをサポートできます。

記録ストレージ

記録ストレージは、**/records** で使用可能なインスタンス全体のストレージデバイスで使用できます。両方の CF VM には、記録ストレージに適したサイズ（最小 16 GB）の 2 番目の vHDD（/hd-raid）がプロビジョニングされます。CF は RAID 設定を共有して、vHDD 間のデータをミラーリングします。SF は、DI ネットワークを介してアクティブ CF にデータレコードを送信し、手動で作成しマウント、または VNFM によってオーケストレーションした外部の一時的なストレージに転送します。

パケットフロー

SF ポートを使用して、ベアラーとシグナリングパケットを送受信します。ネットワーク設定とアドレスの使用をシンプルにするため、すべての SF に接続する必要があるのは、高帯域幅（ベアラー）パケット用の VLAN のみです。低帯域幅インターフェイス（シグナリング）は 2 つの SF のみに接続できます。次の図では、ベアラー VLAN はすべての SF に接続され、シグナリングとその他の VLAN は最初の 2 つの SF のみに接続されています。

（注） この非対称の配置は、必要なインターフェイスの数は少なくなりますが、2 つの VM の損失によりサービスが損失することになるため、障害については慎重に検討する必要があります。

ECMP はハッシュに基づいてハッシングを行って、任意の SF VM にトラフィックを送信できます。

入力では、SF はフロールックアップを実行し、特定の SF 上の特定の SESSMGR タスクにパケットを転送します。この入力トラフィックの一部は、ローカルSESSMGRタスクによって処理されます。それ以外の場合は、DI ネットワークを介して正しい SF にリレーされます。出力では、各 SF がローカルポートからパケットを送信します（ECMP を使用している場合）。ほとんどの場合、パケットが通過する VM の数は 2 未満です。ただし、ACL とトンネリングにより、EPC の設定に応じて、特定のフローのホップ数を増加させる可能性があります。

SF Demux VM で受信したパケット

demux とスタンバイ SF では、受信したすべてのセッショントラフィックが別の SF にリレーされ、セッション処理が行われます。次の図は、入力パケットが Demux SF を介して他のセッション SF にどのように分散されて処理されるかを示しています。

項目	説明
1	受信 NPU はフローのルックアップを行い、SESSMGR を特定します（入力）。
2	SESSMGR はパケットを処理します。
3	送信 NPU はローカルポートにパケットを送信します。（出力）

SF セッション VM で受信したパケット

次の図に、セッション SF が受信した入力パケットをどのように他のセッション SF に配信して処理するかを示します。

項目	説明
1	受信 NPU はフローのルックアップを行い、SESSMGR を特定します（入力）。
2	SESSMGR はパケットを処理します。
3	送信 NPU はローカルポートにパケットを送信します。（出力）

フィーチャ セット

インターフェイスおよびアドレッシング

VPC-DI インスタンス内の各 VM は、1 つの CPU サブシステムを備えた仮想カードとして表されます。これにより、多くの CLI コマンド、ログ、および機能は、ASR 5500 プラットフォームで実行されている StarOS と同様に動作します。

コンテキスト、サービス、プール、インターフェイス、カード、およびポートの StarOS の概念は、ASR 5500 プラットフォームの場合と同様に各 VM に存在します。

VM が起動すると、VM プロファイル内に設定された vNIC が検出され、「仮想イーサネット」タイプのポートに相当する数が StarOS CLI に表示されます。

vNIC の順序を手動で指定するには、起動パラメータファイルの作成を参照してください。

デフォルトでは、システムはハイパーバイザが提示する順序で vNIC インターフェイスを割り当てます。

- CF VM（スロット 1 および 2）
  - 最初に提示されるインターフェイス（1/0 または 2/0）は DI ネットワーク用です。
  - 2 番目に提示されるインターフェイス（1/1 または 2/1）は 管理ネットワーク用です。
- SF VM（スロット 3 ～ 16）
  - 最初に提示されるインターフェイス（*slot*/0）は DI ネットワーク用です。
  - トラフィックインターフェイス *slot*/10 ～ *slot*/21 は、IaaS VLAN 制御トラフィックとデータトラフィック用です。

（注） StarOS は最大 12 個のサービスポートをサポートしますが、実際のポート数はハイパーバイザによって制限される場合があります。

サポートされているハイパーバイザにリストされているインターフェイスと、KVM ブリッジグループまたは VMware vSwitch が VM インターフェイスと一致させる順序で一致していることを確認する必要があります。

（注） ハイパーバイザの CLI や GUI にリストされている vNIC の順序が、ハイパーバイザが VM に提供する方法と同じであることは保証できません。初期設定では、**show hardware** CLI コマンドを使用して、ハイパーバイザ vNIC 設定に表示されている MAC アドレスを調べ、VM によって学習された MAC アドレスと一致させる必要があります。これにより、VM インターフェイスが目的のブリッジグループまたは VMware vSwitch に接続されていることが確認されます。

暗号化

VPC-DI インスタンス内の VM は、ソフトウェアベースの暗号化とパケットのトンネリングを実行します（高いスループットのハードウェアベースのサービスとは異なります）。ベアラーパケットの暗号化を多用するコールモデル、または重要な PKI（公開キーインフラストラクチャ）のキー生成レートを持つコールモデルでは、重要なコンピューティング リソースが必要になる場合があります。

COTS サーバハードウェアが Intel 89xx チップに基づいて Coleto Creek のチップセットを使用している場合、システムはパケットの暗号化と復号化にこのハードウェアチップを自動的に利用します。ただし、システムが暗号化と復号化にハードウェアチップセットを使用するためには、すべてのサービス機能 Vm でこのチップセットを使用する必要があります。

セキュリティ

外部トラフィックのセキュリティは、トンネリング、暗号化、Access Control List（ACL; アクセスコントロールリスト）、コンテキストの分離、およびユーザ認証機能など、既存の StarOS プラットフォームと同様の機能を備えています。CF および SF 上のユーザポートおよびインターフェイスは、StarOS の CLI 設定によって保護されます。

仮想システムでは、データセンター機器の DI ネットワークを介してネットワーク通信が行われるため、お客様側にさらにセキュリティ上の懸念が生じます。

DI ネットワークは、システムネットワークの VLAN のメンバーシップをその特定の VPC-DI インスタンス内の VM に制限することによって、データセンター内の他のホストから分離する必要があります。ネットワークに誤って追加された他のホストによる DI ネットワークへの不正アクセス、またはルータ、スイッチ、またはハイパーバイザの侵害によって、StarOS のセキュリティ対策が中断または回避される可能性があります。このような中断により、障害、サービスの損失や、制御とベアラーパケットの検出が発生する可能性があります。DI ネットワークへのアクセスを適切に保護することは、StarOS の制御の範囲を超えています。

DI ネットワークコンポーネント（CF および SF など）の VM 間の通信は、外部から提供された SSH キーによる認証によってのみ可能になりました。さらに、システムは、DI ネットワーク内のログインに公開/秘密キーベースの SSH 認証を適用します。パスワード、キー、または LI 情報は保存されず、クリアテキストで送信されます。

オペレータが、管理およびベアラーの LI （合法的傍受）などネットワークを物理的に分離する必要がある場合は、センシティブデータを伝送するため、DI ネットワークの物理的な分離も実行する必要があります。仮想環境では、ネットワークの物理的な分離は不可能な場合、または実用的でない場合があります。これらの要件を持つオペレータは、必要に応じて十分な保護を提供することを確認するために、ハイパーバイザとインフラストラクチャを調査する必要があります。

冗長性と可用性

プラットフォーム要件

仮想システムは、システム冗長性と可用性全体の基盤となるハードウェアとハイパーバイザに依存しています。

ハードウェアとハイパーバイザには、次が備わっている必要があります。

- 実際の冗長ハードウェアコンポーネント（電源、ディスクなど）
- 冗長ネットワークパス（デュアルファブリック/NIC、自動フェールオーバー搭載）
- 冗長ネットワークアップリンク（スイッチ、ルータなど）

基盤となるインフラストラクチャ（ホスト、ハイパーバイザ、およびネットワーク）に期待値を超える可用性と信頼性が備わっている場合にのみ、高可用性を実現できます。システムは、実行されている環境と同じ信頼性のみを備えています。

シャーシ間セッションリカバリ（ICSR）は、冗長ではないハードウェア（CPU、メモリ、マザーボード、ハイパーバイザソフトウェアなど）に障害が発生した場合に、可用性を高め、リカバリ時間を短縮するためにも推奨されます。ICSR は、ゲートウェイのセッションレベルでのみ冗長性を提供します。

CF 冗長性

2 つの CF VM は、VPC-DI インスタンスおよびローカルコンテキストポートやローカル管理ポートを制御するために 1:1 冗長化されています。

両方の CF の管理ポート vNIC は相互に 1:1 冗長であり、インフラストラクチャ内の同じ VLAN に配置する必要があります。一度に 1 つの管理ポートのみがアクティブになります。

（注） ホストまたはハイパーバイザに障害が発生した場合に冗長性を確保するために、2 つの CF VM を同じ物理ホスト（サーバまたはブレード）上で実行することはできません。

SF 冗長性

SF VM ごとに、サービスポートのネットワーク接続が提供されます。各 SF には 1 つ以上のポートと関連付けられたインターフェイスがありますが、SF はペアリングされていないため 1:1 の冗長性を提供しません。

SF ポートの冗長性は、ECMP またはサポートされている別の L3 プロトコルを使用して確立する必要があります。

インスタンスに必要な合計スループットは、セッションリカバリが有効になっている N-2 SF を超えないようにする必要があります。これにより、1 つの SF が失敗しても、他の SF が負荷を引き継ぐことができます。サービス IP アドレスにループバックインターフェイスを使用することを強くお勧めします。

SF とピアルータ間のパス障害を検出するために BFD を使用することをお勧めします。これにより、障害が発生した場合に ECMP パスが除外されます。

セッションリカバリが有効になっている場合、1 つの VM が VPN/Demux になり、残りの VM はセッション処理 VM になります。スタンバイ SF は、他の SF に応じて冗長性を提供できます。

（注） 各 SF VM は、ホストまたはハイパーバイザに障害が発生した場合に冗長性を確保するために、別の物理ホスト上で実行する必要があります。

ICSR のサポート

VPC-DI は、StarOS ソフトウェアリリースで ICSR をサポートするサービスの 2 つのインスタンス間でICSR をサポートします。複数のサービスタイプが使用されている場合は、ICSR をサポートするサービスのみが ICSR を使用できます。

ICSR は、サイト/行/ラック/ホストの停止や主要なソフトウェア障害の冗長性をサポートしています。これを行うには、重複しないホストとネットワークインターコネクト上で2つのインスタンスを実行する必要があります。ICSRは、同様に設定されたインスタンス間でのみサポートされます。VPC-DIインスタンスと別のタイプのプラットフォーム（ASR 5500 など）間の ICSR はサポートされていません。

L3 ICSR がサポートされています。

詳細については、このガイドの「シャーシ間セッションリカバリ」の章を参照してください。

ハイパーバイザ要件

VPC-DIは、次のハイパーバイザで実行する資格があります。

- OpenStack ベースの仮想環境
- VMware ESXi
  - バージョン 6.0：リリース 21.8 より前のリリースでサポート
  - バージョン 6.5：リリース 21.8 および 21.9 でサポート
  - バージョン 6.7：リリース 21.10 以降でサポート
- KVM：Red Hat Enterprise Linux 7.4：Only for use with ASR 5700 の展開でのみ使用

Heat テンプレート（OpenStack 用）および OVF/OVA テンプレート（VMWare ESXi 用）は、CF および SF VM 用に提供されています。

VMware vApp は、VPC-DI インスタンスの VM をまとめてバンドルし、簡単に展開できるようします。

OpenStack で VPC-DI を展開するには、Cisco Elastic Services Controller（ESC）を使用することをお勧めします。OpenStack での ESC を使用した VPC-DI のオンボーディングを参照してください。

（注） 提供されるテンプレートからの価格偏差は、期待されるパフォーマンスと信頼性を維持するために、シスコのエンジニアリングによって承認される必要があります。

CF VM 設定

システムでは、各 CF VM が次のように設定されている必要があります。

- 8 vCPU
- 16 GB RAM
- 最初の vNIC は DI ネットワークです。
- 2 番目の vNIC は管理ポートです。
- 最初の vHDD は、起動イメージと設定ストレージです（/flash、非 RAID、4 GB を推奨）。
- 2 番目の vHDD はレコードストレージ用です（オプション）（hd-local1、RAID、最小 16 GB を推奨）

（注） 両方の CF VM が同一に設定されている必要があります。

SF VM 設定

システムでは、各 SF VM が次のように設定されている必要があります。

- 12 個以上の vCPU（vCPU と vRAM のオプション （16 ページ） を参照）。

（注） SF VM は起動せず、vCPU のこの最小要件が満たされていない場合は、次のエラーが報告されます。

```
Found hardware disparate in minimum number of cores, found n
 cores, minimum expected cores are 12.
```

- 32 GB 以上の vRAM（vCPU と vRAM のオプション （16 ページ） を参照）。
- 最初の vNIC は DI ネットワークです。
- 2 番目とそれ以降の vNIC はサービスポートです。システムは最大 12 個の vNIC をサポートしますが、この数はハイパーバイザによって制限される場合があります。
- vHDD は起動イメージ用で、2 GB が推奨されています。

（注） すべての SF VM が同様に設定されている必要があります。VM ハードウェア設定のモニタリングの詳細については、VM ハードウェアの検証を参照してください。

vCPU と vRAM のオプション

CPU は、システム全体とアプリケーションを単独で完全に実行でき、複数の物理 CPU コアの搭載が可能な 1 つの物理コンピュータチップです。仮想コアテクノロジーは、物理コアごとに複数の論理プロセッサ（vCPU）をサポートします。特定のプラットフォーム上でサポートされている vCPU の合計数は、使用可能な物理コアの数と、各コアに実装されている仮想コアテクノロジーのタイプによって異なります。

CF と SF は、多数の vCPU が割り当てられ、それぞれが 1 つのスレッド（命令のシーケンス）VM 内で実行します。プラットフォーム CPU でサポートされている使用可能な vCPU の数がハイパーバイザを介して VM に割り当てることができる vCPU の最大数を超える場合があります。

（注） VM あたりの vCPU の数は、プラットフォーム CPU でサポートされている vCPU の最大数を超えてはなりません。

パフォーマンスを最大化するには、基盤となるハードウェアに合わせて、vCPU または vRAM の数を調整することが望ましい場合があります。SF は vCPU と vRAM のさまざまな組み合わせをサポートしますが、すべての SF がインスタンス内で同じ組み合わせを共有する必要があります。

ソフトウェアは、vCPU の数とその SF の vRAM の量に基づいて、SF の起動時に SF ごとに最適な SESSMGR タスクの数を決定します。

（注） vCPU の数、vRAM のサイズ、または vNIC tタイプ/カウント（ホットプラグ、バルーニングなどを介す）の動的なサイズ変更はサポートされていません。プロビジョニング後にこれらの値を変更する必要がある場合は、すべての VM をシャットダウンして再設定する必要があります。すべての VM上で一度に再設定を実行できます。CPUと RAM が他のインスタンスと一致しなくなるため、一度に 1 つの VM を再設定することはできません。

vNIC のオプション

このリリースでは、次のような vNIC オプションがサポートされています。

- VMXNET3：VMware 用の準仮想 NIC
- VIRTIO：KMV 用の準仮想 NIC
- ixgbe：Intel 10 ギガビット NIC 仮想機能
- enic：Cisco UCS NIC
- SR-IOV：シングルルート I/O の仮想化 ixgbe インターフェイスと enic インターフェイス

vhost-net および vhost-user のサポート

システムは、vhost-net と vhost-user ベースの両方のバックエンドメカニズムと対話できる DPDK ベースのユーザアプリケーションに基づいて Virtio フロントエンドを実装します。vhost-user と vhost-net は、共有メモリベースの、イベント、および割り込み記述子を使用する vhost パラダイムの実装を提供します。DPDK ベースのフロントエンドドライバを vhost-net と vhost-user と組み合わせて使用すると、Linux ブリッジベースのデータパスと比較して、パフォーマンスの高いデータパスを実現できます。

- vhost-user は、ユーザ空間でパケットを完全に処理し、パフォーマンスを引き上げます。システムは DPDK ベースのユーザ空間アプリケーションにフロントエンドを実装しますが、ホストユーザ空間アプリケーションは vhost-user インターフェイスに基づいてバックエンドを実装します。
- vhost-net は、Virtio ネットワーキングのカーネルレベルのバックエンドを提供します。これにより、Virtio パケット処理タスクをユーザ空間 (QEMU プロセス) から移動し、カーネル (vhost-net ドライバ) に移動することで、仮想化のオーバーヘッドを軽減します。これにより、デバイス エミュレーション コードは、ユーザ空間からシステムコールを実行する代わりに、カーネルサブシステムに直接コールすることができます。

  システムは、vhost-user で 1 つのキューをサポートします。

ハードドライブストレージ

必須/フラッシュ（非 RAID）ドライブに追加した場合、システムは仮想マシン（VM）の下で RAID1 をサポートします。VM ごとに、この表に示す SCSI ID と一致する仮想 SCSI ディスクを CF のみで作成できます。最小ディスクサイズは 16 GB を超えている必要があります。

表 *1:* ディスクマッピング

タイプ	/flash（非 RAID）	hd-local1	注意
KVM	SCSI 0:0:0:0	SCSI 0:0:1:0	Raw ディスクの hd-local1 は RAID1 を使用します
VMware	SCSI 0:0:0:0	SCSI 0:0:1:0	Raw ディスクの hd-local1 と hd-remote1 は RAID1 を使用します

記録ストレージ（CDR および UDR）の場合、CF VM には、予想される記録要件を満たすサイズの 2 番目の vHDD をプロビジョニングする必要があります（最小 16 GB）。レコードは、2 番目の vHDD 上の /records に書き込まれます。

DPDK 内部フォワーダ

Intel Data Plane Development Kit（DPDK）は、VPC アーキテクチャの不可欠な部分であり、システムパフォーマンスを向上させるために使用されます。DPDK 内部フォワーダ（IFTASK）は、パケットの入出力操作を担当するソフトウェアコンポーネントであり、Linux カーネルをバイパスすることによって、ユーザ空間でのパケット処理の高速パスを提供します。これはシステムの運用に必要です。CF または SF のインスタンス化時に、DPDK は CPU コアの合計数に応じて、CPU コアの特定の割合を IFTASK に割り当てます。残りの CPU コアはアプリケーションに割り当てられます。

IFTASK で使用されている CPU コアを特定し、それらの使用率を表示するには、**show npu utilization table** コマンドを次のように使用します。

```
[local]mySystem# show npu utilization table

Wednesday July 06 10:53:55 PDT 2017
          -------iftask-------
   lcore      now   5min  15min
--------  ------ ------ ------
  01/0/1      38%    53%    52%
  01/0/2      51%    55%    55%
  02/0/1      66%    72%    68%
  02/0/2      66%    63%    67%
  03/0/1      57%    55%    55%
  03/0/2      51%    47%    45%
  03/0/3      89%    89%    89%
  03/0/4      88%    88%    89%
  04/0/1      67%    59%    58%
  04/0/2      54%    40%    48%
  04/0/3      89%    89%    90%
  04/0/4      90%    89%    89%
  05/0/1      55%    55%    56%
  05/0/2      68%    45%    45%
  05/0/3      90%    89%    89%
  05/0/4      90%    89%    89%
  06/0/1      50%    58%    58%
  06/0/2      24%    24%    25%
  06/0/3      89%    90%    90%
  06/0/4      91%    90%    90%
```

IFTASK コアを使用せずに VM の CPU 使用率を表示するには、**show cpu info** コマンドを使用します。詳細については、**verbose** キーワードを使用してください。

```
[local]mySystem# show cpu info card 6
Tuesday July 05 10:39:52 PDT 2017
Card 6, CPU 0:
  Status                : Active, Kernel Running, Tasks Running
  Load Average          : 7.74, 7.62, 7.54 (9.44 max)
  Total Memory          : 49152M
  Kernel Uptime         : 4D 5H 7M
  Last Reading:
    CPU Usage           : 25.4% user, 7.8% sys, 0.0% io, 0.1% irq, 66.7% idle
    Poll CPUs           : 4 (1, 2, 3, 4)
    Processes / Tasks   : 177 processes / 35 tasks
    Network             : 164.717 kpps rx, 1025.315 mbps rx, 164.541 kpps tx, 1002.149
mbps tx
    File Usage          : 8256 open files, 4941592 available
```

```
    Memory Usage        : 21116M 43.0% used
  Maximum/Minimum:
    CPU Usage           : 32.9% user, 8.9% sys, 0.0% io, 0.4% irq, 59.1% idle
    Poll CPUs           : 4 (1, 2, 3, 4)
    Processes / Tasks   : 184 processes / 36 tasks
    Network             : 178.388 kpps rx, 1270.977 mbps rx, 178.736 kpps tx, 1168.999
mbps tx
    File Usage          : 8576 open files, 4941272 available
    Memory Usage        : 21190M 43.1% used
```

オーケストレーション

VPC-DI インスタンスが展開されると、VPC-DI が実行されている環境のいくつかの期待は、StarOS の制御を超えます。これらのほとんどは、オーケストレーションシステムの要件に分類されます。

- インストールとパラメータの割り当てなどの VPC-DI VM のプロビジョニング：各 VM の設定、接続、および永続ブロックストレージ。
- DI ネットワークが信頼性の要件を確実に満たすようにするための DI ネットワークの L2 プロビジョニング。
- ネットワーク分離のポリシー強制（該当する場合）。
- 冗長性ルールを強制する VM の物理的な配置。
- CPU、RAM、NIC などの物理リソースに便利なモニタリングツールを提供。

VPC-DI インスタンスの展開にオーケストレーションシステムを使用しない場合は、これらの要件を維持する必要があります。ただし、手動で、または他の方法で、強制的に適用する必要があります。VM ハードウェア設定のモニタリングの詳細については、VM ハードウェアの検証を参照してください。

プロビジョニング

VPC-DI インスタンスのプロビジョニングには、次の 2 つのフェーズがあります。

- VM とネットワーク相互接続が作成され、リンクされます。
- VPC-DIインスタンスはサービス用に設定されています。

IaaS 管理者は、サーバをセットアップしてインターコネクトし、ハイパーバイザの VM テンプレートまたはオーケストレーション ソフトウェアを使用して、一連の VM、DI ネットワーク、およびサービスレベル契約（SLA）の要件を満たすための冗長性設定を作成します。

VPC-DI インスタンスを展開するには、オペレータの展開要件に対応する詳細な構成計画が必要です。

ブート シーケンス

StarOS は、事前にインストールされたディスクテンプレートを QCOW2 形式で使用して、各 VM にインストールされます。スロット番号は、ESC および OpenStack によって管理されます。スロット番号は、VM 設定の一部として割り当てられます。スロット番号は、インストール時に自動検出されます。ハイパーバイザからスロット番号を検出できるように、インストーライメージのインストールは完全に自動化されています。詳細については、ソフトウェアインストールおよびネットワーク展開 （23 ページ）を参照してください。

VM の vNIC の設定を制御する方法については、起動パラメータファイルの作成 を参照してください 。

各 VM は再起動し、VPC-DI インスタンスへの参加を試みます。ブートローダーは、自動起動（スクリプト）、ネットワーク起動、または手動起動によってインスタンスを起動します。

仮想 BIOS が完了すると、VM はローカル vHDD から起動し、CFE（共通ファームウェア環境）を実行します。CFE は、インストール中に作成されたパラメータファイルがあるかどうか、vHDD を検索します。このファイルが検出されて正しく解析された場合、CFE は VM のタイプとスロット番号に応じて異なるパスを取得します。いずれの場合も、最初の vNIC がネットワークのインターフェイスになります。

CF ブートシーケンス

CF は、起動シーケンス中に次の機能を実行します。

- 他の CF が動作しているかどうかを確認します（DI ネットワーク経由）。
- 他の CF が動作している場合は、その CF からの起動を試みます。
  - 他の CF からパラメータと起動イメージを取得しようとします。
  - 成功した場合は、起動イメージを転送して実行します。
- 他の CF が動作していないか、または起動に失敗した場合は、個別に起動します。
  - 起動または設定の優先順位については、ローカル vHDD 上で boot.sys ファイルを検索して解析します。
  - ユーザによって（管理ネットワークまたはローカル vHDD を介して）中断されない限り、boot.sys 内の命令により起動を実行します。

CF 上の CFE では、starfile（起動可能イメージ）を CF 管理 vNIC を介してピアから外部 HTTP サーバまたは TFTP サーバへ、あるいは vHDD 上のローカルファイルからダウンロードできます。これは、boot.sys と StarOS の **boot** CLI コマンドによって実行されます。

（注） HTTP と TFTP による起動は、VIRTIO と VMXNET3 インターフェイスタイプでのみサポートされています。

DI ネットワーク上のネットワークプロトコルによって、マスターの CF が決まります。次に、マスターシップが DI ネットワークを介して SF VM に伝達されます。

SF ブートシーケンス

SF がその vHDD から起動します。次に、DI ネットワークを介してアクティブな CF に接続し、正しいソフトウェアバージョンを起動したかどうかを判断します。SF が正しいソフトウェアバージョンを起動しなかった場合は、CF から正しいバージョンを転送し、自身を再起動します。ソフトウェアバージョンが正常に起動すると、ブートシーケンスが完了します。

帯域幅の要件

オペレータ展開ごとに、VPC-DI インスタンスをホストする L2 スイッチ上での帯域幅要件のモデリングが必要です。

主要なベアラートラフィックの他に、DI ネットワークは VM 間でセッションシグナリングと内部制御データも渡します。

内部制御トラフィックは冗長動作中は多くなりますが、通常動作時は大幅に低下します。次の場合に制御トラフィックの使用率が高くなります。

- アクティブ SF VM からスタンバイ SF へのタスクの移行
- スタンバイ SF の起動または再起動
- SF の起動または再起動
- SF またはスタンバイ CF の起動または再起動
- 大量のシグナリングトラフィック（1 秒あたりのコールイベント（CEP）レートが高い）
- CLI またはバルク統計情報の使用率が非常に多い

CEPS レート、設定、および管理操作に応じて、ベアラースループットに関係なく、各 VM は DI ネットワークインターフェイスに負荷をかけます。この負荷はばらつきが大きい可能性がありますが、平均して VM あたり 1 Gbps 未満です（他の VM よりも使用率が高い VM もあります）。

キャパシティ、CEPS、およびスループット

VPC-DI インスタンスのサイジングには、予想されるコールモデルのモデリングが必要です。

最初のソフトウェアバージョンは、最大 2 つの CF と 14 の SF をサポートしています。

サービスタイプの多くは、他よりも多くのリソースを必要としています。パケットサイズ、セッションあたりのスループット、CEPS（コールイベント/秒）レート、IPSec の使用率（サイト間、サブスクライバ、LI）、その他の VM との競合、および基盤となるハードウェアのタイプ（CPU の速度、vCPU の数）によって、ブスクライバの最大有効数がさらに制限されます。同等のハードウェアとハイパーバイザ設定でのコールモデルの認定が必要です。

ソフトウェアベースの送信バッチ処理により、システムパフォーマンスが大幅に向上します。

診断およびモニタリング

VPC-DI は VM 内で実行されるため、ハードウェア診断またはモニタリングは提供されません。ハードウェアセンサーデータ（温度、電圧、メモリエラー）の取得は、ハイパーバイザと外部モニタリングシステムを介して行われます。基盤となる VM の設定を決定するには、VM ハードウェアの検証を参照してください。

VPC-DI は、CLI の **show** コマンド、バルク統計情報、MIB トラップなどの既存のメカニズムを使用して、VM ごとの vCPU、vRAM、および vNIC の使用状況をモニタおよびエクスポートします。ただし、オペレータは、ハイパーバイザのホストごとの物理 CPU、RAM、および NIC の値のモニタリングがより有用であることを確認できる場合があります。

vNIC には可変の最大スループット（たとえば、1 Gbps または 10 Gbps として定義されていない）があるため、スループットのパーセンテージとして使用率をエクスポートするカウンタとバルク統計情報の値がほとんどない場合があります。絶対値（bps）は VM から取得できますが、物理インフラストラクチャの使用率をハイパーバイザから取得する必要があります。これは、最大スループットが固定されているため、パススルー PF NIC には適用されません。

Cisco Prime Analytics

モビリティ向けの Cisco Prime の分析スイートには、VPC-DI インスタンスの拡張性管理が備わっています。

モビリティ向けの Cisco Prime は、次をサポートしています。

- 無線アクセスネットワーク（RAN）バックホールとパケットコア全体の統合されたオペレータワークフロー
- 一元化されたネットワークの可視性と高度なトラブルシューティングおよび診断
- 事前に統合されたネットワーク管理ソフトウェアコンポーネントによる統合に要する時間とリソースの削減

詳細については、シスコのアカウント担当者にお問い合わせください。

StarOS VPC-DI ビルドのコンポーネント

次の StarOS ビルドファイル名タイプが VPC-DI と関連付けられています。

- **.qvpc-di-<version>.iso**：初期のインストールまたはスタートオーバー ISO ファイル。
- **.qvpc-di-<version>.bin**：更新、アップグレード、またはリカバリファイル（すでに実行しているシステム用）。
- **.qvpc-di-template-libvirt-kvm-<version>.tgz**：KVM libvirt テンプレートと ssi_install.sh。

- **.qvpc-di.qcow2.tgz**：KVM QCOW2 ディスクテンプレート。
- **.qvpc-di-template-vmware.tgz**：VMware ファイル。
- **.qvpc-di-template-vmware-<version>.ova**：VMware OVA テンプレート。

ソフトウェアインストールおよびネットワーク展開

このガイドでは、VPC-DI のコンポーネントが正しくインストールされ、市販（COTS）サーバ上の仮想マシン（VM）上で実行されていることを前提としています。詳細については、プロビジョニング （19 ページ）を参照してください。

また、DI ネットワークも、DI ネットワーク （6 ページ） と 帯域幅の要件 （21 ページ） で指定した要件を満たすため、データセンター内でプロビジョニングされている必要があります。

サポートされているオペレーティングシステムおよびハイパーバイザパッケージとプラットフォーム設定の詳細については、シスコの担当者にお問い合わせください。シスコのアドバンスドサービス（AS）グループは、VPC-DI 製品のコンサルティング、インストール、およびネットワーク展開のサービスを提供しています。