VPC-SI 入門

この章では、シスコの仮想化されたパケットコア（シングルインスタンス（VPC-SI）について説明します。VPC-SI は、モバイル市場の新しいアプリケーションとサービスの迅速な展開を可能にする仮想化されたクラウドアーキテクチャのニーズに対応しています。

製品の説明

この章では、StarOS VPC-SI 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-SI は、StarOS を実行している物理的な Cisco ASR 5500 シャーシの動作を、市販の（COTS）サーバー上で実行できる 1 つの仮想マシン（VM）に統合します。各 VPC-SI VM は独立した StarOS インスタンスとして動作し、物理シャーシの管理およびセッション処理機能を組み込みます。

コントロールおよびユーザープレーン分離（CUPS）の VNF 展開の UGP では、CF および関連する SF VNF はコントロールプレーン機能（CPF）を構成します。SAEGW サービスは、Sxa、Sxb、および Sxab（Sxa と Sxa の組み合わせ）インターフェイスにコントロールプレーン機能を提供するために使用されます。

CUPS ベースの VNF 展開では、UPF は SAEGW-U として機能します。これは StarOS の 1 つのスタンドアロンインスタンスを実行する VNFC として展開されます。展開要件に基づいて拡張性を得るために、複数の UPF VNFC を展開できます。

Day-N の設定手順については、『*Ultra Packet Core CUPS Control Plane Administration Guide*』と『*Ultra Packet Core CUPS User Plane Administration Guide*』を参照してください。

仮想化されたモビリティ機能

VPC-SI は、ネットワークのコア内でモビリティ固有のサービスとアプリケーションを実装する、仮想化されたモビリティ機能のセットで構成されています。これらの機能には、次が含まれています。

- モバイルパケットコア
  - LTE MME（モバイル管理エンティティ）、P-GW（PDN ゲートウェイ）、および S-GW（サービングゲートウェイ）
  - GGSN ゲートウェイ GPRS サポートノード
  - SAE-GW System Architecture Evolution ゲートウェイ
  - SGSN サービング GPRS サポートノード（3G のみ）
- モバイル コントロール プレーン PCRF（ポリシーおよび課金ルール機能）、アプリケーション ゲートウェイ、分析、サービス オーケストレーション、抽象化および制御機能
- スモールセルゲートウェイ
  - HNBGW ホームノード B ゲートウェイ
  - HeNBGW 進化型ホームノード B ゲートウェイ
  - SAMOG S2a Mobility over GTP は、単一のサービスノードで CGW（統合アクセスゲートウェイ）と信頼された WLAN AAA プロキシ（TWAP）機能を組み合わせます。

モバイルクラウドネットワーク（MCN）はネットワークインフラストラクチャであり、VPC-SI とその他のサービスが展開される Infrastructure as a Service（IaaS）、オーケストレーション メカニズム、分析メカニズムなどが備わっています。

VM インターコネクト アーキテクチャ

次の図に、VPC-SI でサポートされている基本的な L2/L3 の相互接続を示します。

上の図では、仮想スイッチはハイパーバイザ内に組み込まれ、データセンター全体の SDN L2 機能をサポートしています。仮想スイッチは 802.1Q トランク（VLAN）を使用して他の仮想スイッチと相互接続します。通常、仮想スイッチは動的にロードされたカーネルモジュールです。

スタンドアロンインスタンス

VPC-SI は基本的に、COTS プラットフォーム上の仮想マシン（VM）内で実行されている StarOS です。これは、企業内、リモートサイト内、または顧客のデータセンター内のスタンドアロンの単一 VM として使用できます。また、VPC-SI を大規模なサービス プロバイダー オーケストレーション ソリューションの一部として統合することもできます。

シングル インスタンス アーキテクチャは、キャパシティの低いシナリオに適しています。VPC-SI 仮想ネットワーク機能（VNF）を拡張するには、特定の VNF タイプ（P-GW、S-GW、MME、PCRF、ファイアウォール、NAT など）に対してネットワークレベルを大幅に設定する必要があります。たとえば、新しい VPC-SI P-GW が追加または削除された場合、この情報を使用してさまざまな Diameter ピアを設定する必要があります。この情報によって DNS がプロビジョニングされるか、またはプロビジョニングが解除されます。

VPC-SI は、サポートされているハイパーバイザ KVM（カーネルベースの仮想マシン）と VMware ESXi とのみ連携します。物理的なデバイスの知識はほとんどないか、まったくありません。

通常、VPC-SI はハードウェアまたはハイパーバイザの障害が発生した場合に物理的な冗長性を提供するために、シャーシ間セッションリカバリ（ICSR）ペアに導入する必要があります。

各 VPC-SI VM が StarOS システム全体のロールを担います。VM の外部で公開されている唯一のインターフェイスは、外部管理およびサービストラフィックのインターフェイスです。各 VM は個別に管理されます。

各 VPC-SI VM は次の StarOS 機能を実行します。

- コントローラのタスク
- CLI とロギング用のアウトオブバンド管理（vSerial と vKVM）
- ローカルコンテキスト vpnmgr
- ローカルコンテキスト管理（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
- fastpath と slowpath を介した NPU シミュレーション
- 非ローカルコンテキスト（vNIC、1 ~ 12）。
- セッション処理の逆多重化と vpnmgr
- 暗号処理

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

VM は単一の 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。

StarOS 用に作成されたアプリケーションは、単一の仮想カードをサポートする 1 スロットの仮想シャーシを備えた別のプラットフォームとして VPC-SI を見なします。

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 の概念は既存のプラットフォームの場合と同様に VPC-SI 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。

- VPC-SIはハイパーバイザによって提供される順序でvNICインターフェイスを割り当てます。
  - 最初に提供されるインターフェイスは、VPC-SI 管理用の 1/1 です。
  - 2番目に提供されるインターフェイスは、VPC-SIサービス制御およびデータトラフィック用の 1/10 です。
  - 任意の 3 番目に提供されるインターフェイスは、VPC-SI サービス制御およびデータトラフィック用の 1/11 です。
  - 任意の 4 番目以降のインターフェイスは、1/12、1/13 になります。
- 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は、VPC-SI インターフェイスと一致させる順序で KVM BR グループまたは VMware vSwitch と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 VPC-SI 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、ハイパーバイザのvNIC設定に表示されているMACアドレスを調べ、VPC-SI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、VPC-SIが目的の BR グループまたは vSwitch に接続されていることが確認されます。

暗号化

VPC-SI 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。

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、およびユーザー認証機能など、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上のユーザー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-SI 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、リソースの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。StarOS は、その上部でシステムの冗長性を処理します。

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。

ICSR のサポート

VPC-SI 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。

（注） 2 つの VPC-DI インスタンスは、異なる DI ネットワーク上にある必要があります。

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-SI インスタンスと別のタイプのプラットフォーム間の 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。

IFTASK 冗長性

アクティブスタンバイタスクが強制終了されるか、または停止されると、スタンバイ IFTASK への移行が行われます。この移行プロセスには、スタンバイ IFTASK からアクティブ IFTASK への特定のパラメータの初期化が含まれます。現在のハートビート間隔では、初期化に3秒以上かかる場合があり、カードが再起動します。

カードの再起動を回避するには、次の設定を使用して HAT HB 間隔を高い値に設定します。

```
configure
  [ default ] high-availability fault-detection card hb-loss  value
```

ハイパーバイザ要件

VPC-SI は、次のハイパーバイザで実行する資格があります。

- カーネルベースの仮想マシン（KVM）：QEMU エミュレータ 2.0 VPC-SI StarOS インストールビルドには、Ubuntu Server14.04 下での VM 作成用の libvirt XML テンプレートと ssi_install.sh が含まれています。
- KVM：Red Hat Enterprise Linux 7.2：VPC-SI StarOS インストールビルドには、qvpc-si_install.sh というインストールスクリプトが含まれています。
- VMware ESXi 6.0：VPC-SI StarOS インストールビルドには、ESXi GUI を介した VM の作成用の OVF（オープン仮想化フォーマット）と OVA（オープン仮想アプリケーション）のテンプレートが含まれています。
- VMware ESXi
  - バージョン 6.0：VPC-SI StarOS インストールビルドには、ESXi GUI を介した VM の作成用のOVF（オープン仮想化形式）およびOVA（オープン仮想アプリケーション）テンプレートが含まれています。このバージョンはリリース21.8より前のリリースでサポートされています。
  - 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

VM の設定

VPC-SI では、VM が次のように設定されている必要があります。

- X vCPU（vCPU と vRAM のオプション （7 ページ） を参照）
- Y vRAM（vCPU と vRAM のオプション （7 ページ） を参照）
- 最初の vNIC は管理ポート（vNIC のオプション （8 ページ） を参照）
- 2 番目以降の vNIC はサービスポートです。1 つの vNIC を必要とし、最大 12 個が VPC でサポートされますが、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。
- 最初の vHDD はブートイメージと設定ストレージ用です（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）（16 GB 以上）

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。

VM が NUMA ノードにまたがらないようにすることを推奨します。vCPU は基盤となる CPU に合わせます。パフォーマンスを最適化するために、VM が NUMA ノードにまたがらないようにします。numa ノードの数の制限は 2 です。

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

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
VMware	SCSI 0:0:0:0	SCSI 0:0:1:0	Raw ディスクの hd-local1 と hd-remote1 は R 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-SI アーキテクチャの不可欠な部分であり、8 個以上の vCPU で設定された VPC-SI システムの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、ユーザー空間でのパケット処理の高速パスを提供します。VPC-SI ブートプロセス中に、vCPU の比率が IFTASK に割り当てられ、残りの部分はアプリケーション処理に割り当てられます。

IFTASK で使用される vCPU 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

******** show npu utilization table card 1 *******
                5-Sec Avg:
lcore00|lcore01|lcore02|lcore03|lcore04|lcore05|lcore06|lcore07|
                    IDLE:          |     4%|    43%|    24%|    45%|    17%|        |
     |
           QUEUE_PORT_RX:          |    34%|    56%|        |        |        |        |
     |
           QUEUE_PORT_TX:          |        |        |        |        |        |        |
     |
           QUEUE_VNPU_RX:          |        |        |        |        |        |        |
     |
           QUEUE_VNPU_TX:          |        |        |        |        |        |        |
     |
            QUEUE_KNI_RX:          |    57%|        |        |        |        |        |
     |
            QUEUE_KNI_TX:          |        |        |        |        |        |        |
     |
        QUEUE_THREAD_KNI:          |     3%|        |        |        |        |        |
     |
          QUEUE_MCDMA_RX:          |        |        |     7%|        |     5%|        |
     |
          QUEUE_MCDMA_TX:          |        |        |        |        |        |        |
     |
       QUEUE_THREAD_MCDMA:         |        |        |     8%|        |    24%|        |
     |
         QUEUE_THREAD_VNPU:        |        |        |        |        |        |        |
     |
           QUEUE_CRYPTO_RX:        |        |        |        |        |        |        |
     |
           QUEUE_CRYPTO_TX:        |        |        |        |        |        |        |
     |
        QUEUE_THREAD_IPC:          |        |        |        |        |        |        |
     |
             MCDMA_FLUSH:          |        |        |    59%|    54%|    51%|        |
     |
    QUEUE_THREAD_TYPE_MAX:         |        |        |        |        |        |        |
      |
                300-Sec Avg:
lcore00|lcore01|lcore02|lcore03|lcore04|lcore05|lcore06|lcore07|
                    IDLE:          |    99%|   100%|    31%|    30%|    32%|        |
     |
           QUEUE_PORT_RX:          |     0%|        |        |        |        |        |
     |
           QUEUE_PORT_TX:          |        |        |        |        |        |        |
     |
           QUEUE_VNPU_RX:          |        |        |        |        |        |        |
     |
           QUEUE_VNPU_TX:          |        |        |        |        |        |        |
     |
            QUEUE_KNI_RX:          |        |        |        |        |        |        |
     |
            QUEUE_KNI_TX:          |        |        |        |        |        |        |
     |
        QUEUE_THREAD_KNI:          |        |        |        |        |        |        |
```

```
|
      QUEUE_MCDMA_RX:          |       |       |     0%|     0%|     0%|       |
|
      QUEUE_MCDMA_TX:          |       |       |       |       |       |       |
|
  QUEUE_THREAD_MCDMA:          |       |       |       |       |       |       |
|
   QUEUE_THREAD_VNPU:          |       |       |       |       |       |       |
|
     QUEUE_CRYPTO_RX:          |       |       |       |       |       |       |
|
     QUEUE_CRYPTO_TX:          |       |       |       |       |       |       |
|
    QUEUE_THREAD_IPC:          |       |       |       |       |       |       |
|
         MCDMA_FLUSH:          |       |       |    68%|    69%|    67%|       |
|
QUEUE_THREAD_TYPE_MAX:         |       |       |       |       |       |       |
|
          900-Sec Avg:
lcore00|lcore01|lcore02|lcore03|lcore04|lcore05|lcore06|lcore07|
               IDLE:           |    99%|   100%|    31%|    31%|    32%|       |
|
       QUEUE_PORT_RX:          |     0%|       |       |       |       |       |
|
       QUEUE_PORT_TX:          |       |       |       |       |       |       |
|
       QUEUE_VNPU_RX:          |       |       |       |       |       |       |
|
       QUEUE_VNPU_TX:          |       |       |       |       |       |       |
|
        QUEUE_KNI_RX:          |       |       |       |       |       |       |
|
        QUEUE_KNI_TX:          |       |       |       |       |       |       |
|
    QUEUE_THREAD_KNI:          |       |       |       |       |       |       |
|
      QUEUE_MCDMA_RX:          |       |       |     0%|     0%|     0%|       |
|
      QUEUE_MCDMA_TX:          |       |       |       |       |       |       |
|
  QUEUE_THREAD_MCDMA:          |       |       |       |       |       |       |
|
   QUEUE_THREAD_VNPU:          |       |       |       |       |       |       |
|
     QUEUE_CRYPTO_RX:          |       |       |       |       |       |       |
|
     QUEUE_CRYPTO_TX:          |       |       |       |       |       |       |
|
    QUEUE_THREAD_IPC:          |       |       |       |       |       |       |
|
         MCDMA_FLUSH:          |       |       |    68%|    68%|    67%|       |
|
QUEUE_THREAD_TYPE_MAX:         |       |       |       |       |       |       |
|


thread 2 QUEUE_PORT_RX                  77.22 %
thread 2 IDLE                           22.78 %
------------------------------------------------
thread 5 MCDMA_FLUSH                    57.74 %
thread 5 IDLE                           40.13 %
thread 5 QUEUE_THREAD_MCDMA              2.13 %
```

```
-------------------------------------------------
thread 1 QUEUE_KNI_RX                    50.39 %
thread 1 QUEUE_PORT_RX                   40.72 %
thread 1 IDLE                             6.13 %
thread 1 QUEUE_THREAD_KNI                 2.76 %
-------------------------------------------------
thread 3 QUEUE_THREAD_MCDMA              41.17 %
thread 3 MCDMA_FLUSH                     38.31 %
thread 3 IDLE                            16.28 %
thread 3 QUEUE_MCDMA_RX                   4.24 %
-------------------------------------------------
thread 4 IDLE                            56.03 %
thread 4 MCDMA_FLUSH                     43.97 %
-------------------------------------------------
```

IFTASK vCPU 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 1, CPU 0:
  Status                : Active, Kernel Running, Tasks Running
  Load Average          : 8.99, 9.50, 8.20 (11.89 max)
  Total Memory          : 16384M
  Kernel Uptime         : 0D 0H 49M
  Last Reading:
    CPU Usage           : 16.6% user, 10.5% sys, 0.0% io, 4.6% irq, 68.3% idle
    Poll CPUs           : 5 (1, 2, 3, 4, 5)
    Processes / Tasks   : 234 processes / 54 tasks
    Network             : 353.452 kpps rx, 3612.279 mbps rx, 282.869 kpps tx, 2632.760
mbps tx
    File Usage          : 2336 open files, 1631523 available
    Memory Usage        : 4280M 26.1% used, 42M 0.3% reclaimable
  Maximum/Minimum:
    CPU Usage           : 23.2% user, 11.2% sys, 0.1% io, 5.5% irq, 61.5% idle
    Poll CPUs           : 5 (1, 2, 3, 4, 5)
    Processes / Tasks   : 244 processes / 54 tasks
    Network             : 453.449 kpps rx, 4635.918 mbps rx, 368.252 kpps tx, 3483.816
mbps tx
    File Usage          : 3104 open files, 1630755 available
    Memory Usage        : 4318M 26.4% used, 46M 0.3% reclaimable
```

キャパシティ、CEPS、およびスループット

VPC-SI 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。

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。

診断およびモニタリング

VPC-SI はVM内で実行されるため、ハードウェアの診断またはモニタリングは行われません。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。

VPC-SI は、CLI **show** コマンド、バルク統計情報、およびMIB トラップなどの既存のメカニズムを通じて vCPU、vRAM、および vNIC の使用状況をモニターし、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内の物理 CPU、RAM、および NIC の値のモニタリングのほうが有用であると気づく場合があります。

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VPC-SI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るため、パススルー PF NIC には適用されません。

StarOS VPC-SI のビルドコンポーネント

次の StarOS ビルドのファイル名のタイプは、次のように VPC-SI と関連付けられています。

- **qvpc-si-<version>.iso**：初期インストールまたはスタートオーバー ISO ファイル
- **qvpc-si-<version>.bin**：すでに実行されているシステムの更新、アップグレード、またはリカバリファイル。詳細については、「*StarOS* 管理操作」の章を参照してください。
- **qvpc-si-template-libvirt-kvm-<version>.tgz**：KVM libvirt テンプレートと ssi_install.sh
- **qvpc-si-template-vmware-<version>.ova**：VMware OVA テンプレート
- **qvpc-si-<version>.qcow2.gz**：KVM QCOW2 ディスクテンプレート

VPC-SI ブートパラメータ

起動パラメータファイルは、起動する前にStarOSに設定項目を渡すための手段を提供します。これらのパラメータは、管理、サービス、および VNFM インターフェイスの詳細などの項目と StarOS の起動時に作成された内部フォワーダータスク（iftask）の設定を指定します。

起動パラメータは複数の方法で送信され、すべての方式が同じパラメータ名と使用法を使用します。ブートパラメータファイルの最初の場所は、最初の VM ドライブ（*/boot1/param.cfg* など）の最初のパーティションになります。検索された 2 番目の場所は、仮想 CD-ROM ドライブである設定ドライブ上にあります。OpenStack を使用している場合は、ターゲットの起動パラメータのファイル名を *staros_param.cfg* として指定します。OpenStack を使用していない場合は、ルートディレクトリに *staros_param.cfg* を使用して ISO イメージを作成し、この ISO を VM の最初の仮想 CD-ROM ドライブに接続します。

VM が起動すると、最初に CFE と呼ばれるブート前環境で param.cfg ファイルが解析されます。VM が Linux を開始すると、仮想 CD-ROM ドライブにアクセスして、 staros_param.cfg ファイルを解析します。*/boot1/param.cfg* ファイルに保存されている値との競合がある場合は、staros_param.cfg のパラメータが優先されます。

起動パラメータファイルのフォーマット

起動パラメータファイルの構造は次のとおりです。

VARIABLE_NAME = VALUE

行区切りとして改行を使用して、1 行につき 1 つの変数を指定します（UNIX テキストファイル形式）。変数名と値は大文字と小文字が区別されません。無効な値は無視され、エラー通知が VM コンソールに表示されます。変数の値が重複している場合（同じ変数名に 2 つの異なる値が指定されている場合）、最後に定義された値が使用されます。

数値にゼロを埋める必要はありません。たとえば、PCI_ID は 0:1:1.0 の場合、0000:01:01.0 と同様に扱われます。

ネットワーク インターフェイス ID

VPC-SIハイパーバイザによって提供される順序で vNIC インターフェイスを割り当てます。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。

VPC-SI が vNIC を検出する順序は PCI バスの列挙順に従い、準仮想デバイスも PCI バスで表されます。PCI バスは、同じレベルの追加デバイスの前にブリッジが探索される深さ優先の方法で列挙されます。すべてのネットワーク インターフェイスのタイプが同じ場合、PCI トポロジを認識するだけで vNIC の正しい順序を取得できます。ネットワーク インターフェイスのタイプが異なる場合、その順序は PCI トポロジに加えて VM 内のデバイスドライバのロード順序に依存します。デバイスドライバのロード順序は、ソフトウェアのリリース順と同じである保証はありませんが、一般的には準仮想デバイスがパススルーデバイスよりも優先されます。

NIC を識別するために使用できる方式はいくつかあります。

- MAC アドレス：インターフェイスの MAC アドレス
- 仮想 PCI ID
- 結合インターフェイス：ネットワークデバイスのボンディングを使用すると、ネットワーク インターフェイスはスレーブ インターフェイス ロールとして機能するように識別されます。ボンドのスレーブインターフェイスは、MAC、PCI ID、またはインターフェイスタイプを使用して識別されます。
- インターフェイスタイプおよびインスタンス番号

仮想 PCI ID

PCI バス上のデバイスは、ドメイン、バス、デバイス、および機能番号と呼ばれる一意のタプルによって識別されます。これらの識別子は、いくつかの方法で識別できます。

ゲスト内では、**lspci** ユーティリティによって次のようにバスの設定が表示されます。

```
# lspci

00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371SB PIIX3 IDE [Natoma/Triton II]
00:01.2 USB controller: Intel Corporation 82371SB PIIX3 USB [Natoma/Triton II] (rev 01)
00:01.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 03)
00:02.0 VGA compatible controller: Cirrus Logic GD 5446
00:03.0 System peripheral: Intel Corporation 6300ESB Watchdog Timer
00:04.0 Unclassified device [00ff]: Red Hat, Inc Virtio memory balloon
00:05.0 Ethernet controller: Red Hat, Inc Virtio network device
00:06.0 Ethernet controller: Red Hat, Inc Virtio network device
```

この仮想バスのドメイン、バス、デバイス、および機能番号を次に示します。

表 *2 :* 仮想 *PCI ID*

回線	ドメイン	バス	デバイス	機能
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)	0	0	0	0
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]	0	0	1	0
00:01.1 IDE インターフェイス：Intel Corporation 82371SB PIIX3 IDE [Natoma/Triton II]	0	0	1	1
00:01.2 USB コントローラ：Intel Corporation 82371SB PIIX3 USB [Natoma/Triton II] (rev 01)	0	0	1	2
00:01.3 ブリッジ：Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 03)	0	0	1	3
00:02.0 VGA 互換コントローラ：Cirrus Logic GD 5446	0	0	2	0
00:03.0 システム周辺機器：Intel Corporation 6300ESB ウォッチドッグタイマー	0	0	3	0
00:04.0 未分類のデバイス [00ff]：Red Hat、Inc Virtio メモリバルーニング	0	0	4	0
00:05.0 イーサネットコントローラ：Red Hat、Inc Virtio ネットワークデバイス	0	0	5	0

回線	ドメイン	バス	デバイス	機能
00:06.0 イーサネットコントローラ：Red Hat、Inc Virtio ネットワークデバイス	0	0	6	0

libvirt ベースの仮想マシンの場合は、**virsh dumpxml** コマンドから仮想 PCI バストポロジを取得できます。libvirt スキーマでは、デバイス番号に *slot* という用語が使用されることに注意してください。これは、前の例で使用した仮想マシンにおける xml の説明のスニペットです。

```
<interface type='bridge'>
  <mac address='52:54:00:c2:d0:5f'/>
  <source bridge='br3043'/>
  <target dev='vnet0'/>
  <model type='virtio'/>
  <driver name='vhost' queues='8'/>
  <alias name='net0'/>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x05' function='0x0'/>
</interface>
<interface type='bridge'>
  <mac address='52:54:00:c3:60:eb'/>
  <source bridge='br0'/>
  <target dev='vnet1'/>
  <model type='virtio'/>
  <alias name='net1'/>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x06' function='0x0'/>
</interface>
```

インターフェイスタイプおよびインスタンス番号

ここで NIC は、Linux デバイスドライバの名前（virtio_net、vmxnet3、ixgbe、i40e など）とそのインスタンス番号を使用して、そのタイプによって識別されます。インスタンス番号は、そのタイプのインターフェイスの PCI 列挙順に基づいています。インスタンス番号 1 から始まります。インターフェイスタイプは、パススルーインターフェイスと SR-IOV 仮想機能だけでなく、両方の準仮想タイプを識別するために使用できます。PCI バス上のデバイスの PCI 列挙の順序は、ホスト OS 上の**lspci**ユーティリティで確認できます。

たとえば、次のゲスト PCI トポロジの CF は、virtio_net インターフェイス番号 1 が 00:05.0 のイーサネットコントローラであり、virtio_net インターフェイス番号 2 が 00:06.0 のイーサネットコントローラであることを示しています。出力は、ゲストで実行された **lspci** コマンドからのものです。

```
# lspci

00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371SB PIIX3 IDE [Natoma/Triton II]
00:01.2 USB controller: Intel Corporation 82371SB PIIX3 USB [Natoma/Triton II] (rev 01)
00:01.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 03)
00:02.0 VGA compatible controller: Cirrus Logic GD 5446
00:03.0 System peripheral: Intel Corporation 6300ESB Watchdog Timer
00:04.0 Unclassified device [00ff]: Red Hat, Inc Virtio memory balloon
00:05.0 Ethernet controller: Red Hat, Inc Virtio network device
```

```
00:06.0 Ethernet controller: Red Hat, Inc Virtio network device
```

サポートされている Linux ドライバの完全なリストを次に示します。

表 *3*: サポートされている *Linux* ドライバ

タイプ	PCI ベンダー/デバイス ID	ドライバ名
VIRTIO（KVM 用の準仮想 NIC）	0x10af / 0x1000	virtio_net
VMXNET3（VMware 用の準仮想 NIC）	0x15ad / 0x07b0	vmxnet3
Intel 10 ギガビットイーサネット	0x8086 / 0x10b6 0x8086 / 0x10c6 0x8086 / 0x10c7 0x8086 / 0x10c8 0x8086 / 0x150b 0x8086 / 0x10dd 0x8086 / 0x10ec 0x8086 / 0x10f1 0x8086 / 0x10e1 0x8086 / 0x10db 0x8086 / 0x1508 0x8086 / 0x10f7 0x8086 / 0x10fc 0x8086 / 0x1517 0x8086 / 0x10fb 0x8086 / 0x1507 0x8086 / 0x1514 0x8086 / 0x10f9 0x8086 / 0x152a 0x8086 / 0x1529 0x8086 / 0x151c 0x8086 / 0x10f8 0x8086 / 0x1528 0x8086 / 0x154d 0x8086 / 0x154f 0x8086 / 0x1557	ixgbe

タイプ	PCI ベンダー/デバイス ID	ドライバ名
Intel 10 ギガビット NIC 仮想機能	0x8086 / 0x10ed 0x8086 / 0x1515	ixgbevf
Cisco UCS NIC	0x1137 / 0x0043 0x1137 / 0x0044 0x1137 / 0x0071	enic
Mellanox ConnectX-5 （注） • Mellanox は、ユーザープレーンでサポートされています。 • Mellanox は、VPC-DI の管理インターフェイスでサポートされています。	0x15b3 / 0x1017 0x15b3 / 0x1018	mlx5_core
Intel XL 710 ファミリ NIC（PF）	0x8086 / 0x1572（40 ギガ） 0x8086 / 0x1574（40 ギガ） 0x8086 / 0x1580（40 ギガ） 0x8086 / 0x1581（40 ギガ） 0x8086 / 0x1583（40 ギガ） 0x8086 / 0x1584（40 ギガ） 0x8086 / 0x1585（40 ギガ） 0x8086 / 0x158a（25 ギガ） 0x8086 / 0x158b（25 ギガ）	i40e**
Intel XL 710 ファミリ NIC 仮想機能	0x8086 / 0x154c	i40evf

* * 注：**i40e**ドライバの使用時に、ホスト上で作成された SRIOV VF に対して MAC アドレスの割り当てが動的に行われないという既知の問題が存在します。StarOS VM を起動するには、MACアドレスの割り当てが必要です。回避策として、MACアドレスの割り当てはホストから設定する必要があります。詳細は次のリンクを参照してください。https://www.intel.com/content/dam/www/public/us/en/documents/technology-briefs/xl710-sr-iov-config-guide-gbe-linux-brief.pdf

VPC-SI ブートパラメータの設定

VPC-SI VM には、管理ネットワークに接続するために 1 つのインターフェイスが設定されています。このインターフェイスは通常、StarOS で設定され、Day 0 設定に含まれている必要があります。管理インターフェイスは、メインの StarOS 設定ファイルを使用してスタティックアドレスの割り当てをサポートします。

0 ～ 4 個の追加ネットワーク インターフェイスはサービスポートとして機能します。これらのインターフェイスは、StarOS によって設定されます。通常、これらのポートは VNF インフラストラクチャ（VNFI）のトランクポートとして設定されます。

仮想ネットワーク機能（VNF）マネージャ（VNFM）が存在する場合は、それに接続されているネットワーク インターフェイスを設定するオプションが VPC-SI VM に備わっています。このインターフェイスは、DHCP またはスタティック IP 割り当てを使用して設定でき、VNFM 以上のレベルのオーケストレータとの通信に使用されます。このインターフェイスは、メインアプリケーションが起動する前に有効になります。

表 *4: VPC-SI* ブートパラメータ

パラメータ	説明
MGMT_INTERFACE=*value*	VPC-SI VM 上の管理ポートへのインターフェイス有効な値は次のとおりです。 • MAC: xx:xx:xx:xx:xx:xx • PCI_ID: xxx:xx:xx.x (Domain:Bus:Device.Function) • TYPE：*drive-name-instance-number* • BOND：*slave-interface-A,slave-interface-B* インターフェイス識別子の決定については、ネットワーク インターフェイス ID （13 ページ）を参照してください。
SERVICE#_INTERFACE=*value*	サービスポート番号 # へのインターフェイス（# は 1 ～ 4）。 サービスインターフェイスは、**MGMT_INTERFACE** についての上記と同じ値をサポートしています。
SERVICE_INTERFACE_MTU=*value*	デフォルトでは、IFTASK プロセスは最大インターフェイス MTU を 2,100 バイトに設定します。 *value* は 576 ～ 9,100 バイトの整数にする必要があります。

パラメータ	説明
VNFM_INTERFACE=*value*	仮想ネットワーク機能マネージャ（VNFM）またはオーケストレータへのオプションのネットワークインターフェイス。 VNFM インターフェイスでは、**MGMT_INTERFACE** についての上記と同じ値がサポートされています。
VNFM_IPV4_ENABLE=*{true | false }*	VNFM インターフェイスを有効にします。
VNFM_IPV4_DHCP_ENABLE=*{true | false }*	VNFM に対して DHCP を有効にします。
VNFM_IPV4_ADDRESS=*value*	DHCP が使用されていない VNFM の IP アドレスを指定します。
VNFM_IPV4_NETMASK=*value*	DHCP が使用されていない VNFM の IP アドレスのネットマスクを指定します。
VNFM_IPV4_GATEWAY=*value*	DHCP が使用されていない VNFM の IP アドレスのゲートウェイを指定します。
IFTASK_CRYPTO_CORES=*value*	（任意） （任意）IFTASK_CRYPTO_CORES IFTASK_SERVICE_TYPE が「2」（EPDG）に設定されている場合、このパラメータは iftask コアの 暗号処理に割り当てるパーセンテージを指定します。値の範囲は 0 ～ 50 パーセントですが、専用のコアの上限は 4 です。 デフォルトは 0 です。 このパラメータは IFTASK_SERVICE_TYPE が「2」（EPDG）に設定されている場合にのみ使用する必要があります。他のサービスタイプに 設定されている場合、このパラメータは「0」に設定する必要があります。

パラメータ	説明
IFTASK_MCDMA_CORES=*value*	（任意）マルチチャネルダイレクトメモリアクセス（MCDMA）に割り当てられたコア数をiftaskコアの総数の割合として設定します。まず、上記のIFTASK_CORESパラメータを定義する必要があります。注：NUMA最適化が有効になっており、このMCDMAコアカウント設定も構成されている場合は、IFTASK_CORES=percentage-of-coresを偶数に設定する必要があります。これにより、MCDMAスレッドがNUMAノード間で均等に分散されます。
CHASSIS_ID=*value*	シャーシIDは、CLI設定（管理者クレデンシャル、SNMPコミュニティ、RADIUS/Diameterデータ、認証データなど）でselectコマンドを保護します。設定を保存またはロードする必要があり、通常は、CLIコマンドからランダムなデータとユーザーシードを使用して生成されます。その後、シャーシIDはシステムの「/flash」ストレージに保存され、後で保存された設定をロードするために使用されます。 このオプションを設定すると、/flashのパラメータコンフィギュレーションファイルからシャーシIDがロードされます。/flashのシャーシIDは無視され（以前のバージョンからのアップグレードをサポート）、CLIコマンドを使用した新しいシャーシIDの生成は禁止されます。この方法を使用して指定した場合、シャーシIDは/flashに保存されなくなります。また、この操作では、ESC（または別のVNFM）でVMを管理し、シャーシIDを指定する必要もあります。

パラメータ	説明
IFTASK_SERVICE_TYPE=*value*	（任意）サービスメモリを計算しサービス固有の機能を有効にするために展開するサービスタイプを指定します。デフォルトは0です。 次のサービスタイプを指定できます。 • 0 = VPC サービスタイプ • 1 = GiLAN サービスタイプ • 2 = ePDG サービスタイプ • 3 = CUPS コントローラサービスタイプ • 4 = CUPS フォワーダサービスタイプ
FORWARDER_TYPE=*value*	フォワーダタイプを「vpp」または「iftask」として指定します。 デフォルトでは、フォワーダタイプは「iftask」です。フォワーダタイプは、FORWARDER_TYPE が VPP として適切に設定されている場合にのみ、VPP になります。 例：**FORWARDER_TYPE=***vpp*

params.cfg の例

```
MGMT_INTERFACE=MAC:00:01:02:03:04:05
SERVICE1_INTERFACE=PCI_ID:0000:01:02.0
SERVICE2_INTERFACE=PCI_ID:0000:01:03.0
VNFM_INTERFACE=MAC:00:01:02:03:04:AA
VNFM_IPV4_ENABLE=true
VNFM_IPV4_DHCP_ENABLE=false
VNFM_IPV4_ADDRESS=10.1.1.100
VNFM_IPV4_NETMASK=255.255.255.0
VNFM_IPV4_GATEWAY=10.1.1.1
IFTASK_MCDMA_CORES=20
CHASSIS_ID=2sg9x1wqbj
IFTASK_SERVICE_TYPE=0
```

StarOS コマンド **show cloud hardware iftask** を使用して iftask ブートパラメータが有効になったことを確認します。

VNFM インターフェイスオプションの設定

（注） これらの設定オプションは任意です。

仮想ネットワーク機能マネージャ（VNFM）インターフェイスは、各 VM と VNFM の間で通信するように設計されています。VNFMインターフェイスはメインアプリケーションの前に初期化するため、インターフェイスを設定できるのは起動パラメータのみとなります。

デフォルトでは、VNFM インターフェイスは無効になっています。

VNFM IPv4 インターフェイスの有効化

デフォルト値は [False]（無効）です。

変数	有効な値
VNFM_IPV4_ENABLE	True または False

IPv4 DHCP クライアントの設定

変数	有効な値
VNFM_IPV4_DHCP_ENABLE	True または False

IPv4 スタティック IP の設定

（注） IPv4 DHCP クライアントが有効になっている場合、スタティック設定パラメータは無視されます。

変数	有効な値
VNFM_IPV4_ADDRESS	x.x.x.x
VNFM_IPV4_NETMASK	x.x.x.x
VNFM_IPV4_GATEWAY	x.x.x.x

VNFM IPv6 インターフェイスを有効にします。

変数	有効な値
VNFM_IPV6_ENABLE	True または False

IPv6 スタティック IP 設定の有効化

変数	有効な値
VNFM_IPV6_STATIC_ENABLE	True または False

True に設定すると、次の項に示すように、スタティック IP パラメータの設定がインターフェイスに適用されます。False に設定すると、インターフェイスはステートレス自動設定（RFC4862）と DHCPv6 の両方を使用してインターフェイスのアドレスを設定しようとします。

IPv6 スタティック IP の設定

（注） 「VNFM_IPV6_ENABLE」パラメータ値が false に設定されている場合、スタティック設定パラメータは無視されます。IPv6 アドレスフィールドは、RFC 5952 に準拠している必要があります。プレフィックスは /64 で固定されています。

変数	有効な値
VNFM_IPV6_ADDRESS	x:x:x:x:x:x:x:x
VNFM_IPV6_GATEWAY	x:x:x:x:x:x:x:x

VPP 設定パラメータ

以降の項では、FORWARDER_TYPE に VPP を選択した場合にのみ適用されるパラメータを示します。これらのパラメータを使用すると、VPP の CPU とインターフェイス設定をきめ細かく制御できます。

（注） どのような VPP 設定パラメータをオーバーライドする場合も、事前にシスコのアカウント担当者に連絡してオーバーライド値を確認してください。

VPP の CPU 割り当て

VPP ワーカーは、CPU コア全体を消費するリアルタイムのスレッドです。VPP のメインスレッドはコア全体を消費しませんが、ビジー状態になる可能性があります。したがって、競合を回避するために割り当てます。

次の表に、VPP-CPU パラメータを示します。

表 *5 : VPP-CPU* パラメータ

パラメータ	説明
VPP_CPU_MAIN=*value*	Linux プロセッサ番号を指定します。これは 0 ～（CPU の数から 1 を差し引いた数）になります。デフォルト値は 1 です。 次の例を使用して、メインスレッドの値を Linux プロセッサ番号 1 に設定します。 `VPP_CPU_MAIN=1`
VPP_CPU_WORKER_CNT=*value*	Linux プロセッサに設定されているワーカースレッドの数を指定します。有効な値は 0 ～（CPU の数から 3 を差し引いた数）です。デフォルト値は、Linux CPU の 50 %、または VPP_CPU_WORKER_LIST 内の CPU の数です。 ワーカースレッドの数を 3 に設定するには、次の例を使用します。 `VPP_CPU_WORKER_CNT=3`
VPP_CPU_WORKER_LIST=*value*	Linux プロセッサに設定されたワーカースレッドを指定します。ワーカーリストは、Linux プロセッサ番号をカンマで区切ったリストです。有効な値は 0 ～（CPU の数から 1 を差し引いた数）です。デフォルトは、すべてのソケットに割り当てられるラウンドロビン番号であり、すべてのソケットの最初のコアと最初のソケットの 2 番目のコアをスキップします。 次の例を使用して、ワーカースレッドの数を Linux プロセッサ 数 2、9、および 10 に設定します。 `VPP_CPU_WORKER_LIST=2,9,10`

デフォルトの DPDK 設定

表に示す次のパラメータは、一般的に DPDK を設定するか、またはインターフェイスのデフォルトを設定します。

表 *6 : DPDK* パラメータ

パラメータ	説明
VPP_DPDK_BUFFERS=*value*	DPDKバッファの数を指定します。最小バッファは 32,000 で、最大値は VM サイズに基づきます。デフォルトのバッファ数は 128,000 です。 DPDK バッファを 200,000 に設定するには、次の例を使用します。 `VPP_DPDK_BUFFERS=200000`
VPP_DPDK_RX_QUEUES=*value*	特定の設定がないすべてのインターフェイスの RX キューの数を指定します。有効な値の範囲は、インターフェイスのタイプとホストの設定に応じて 1 ～ 64 になります。デフォルト値は VPP_CPU_WORKER_COUNT から計算され、各ワーカーに少なくとも 1 つのキューが割り当てられた状態でキューの全体的な数を最小限に抑えます。 次の例を使用して、デフォルトの RX キューの数を 2 に設定します。 （注） 一致しない値は、インターフェイスのタイプとホストの設定に基づいて設定できます。 `VPP_DPDK_RX_QUEUES=2`
VPP_DPDK_TX_QUEUES=*value*	特定の設定がないすべてのインターフェイスの TX キューの数を指定します。有効な値の範囲は、インターフェイスのタイプとホストの設定に応じて 1 ～ 64 になります。デフォルト値は VPP_DPDK_RX_QUEUES です。 次の例を使用して、デフォルトの TX キューの数を 4 に設定します。 （注） 一致しない値は、インターフェイスのタイプとホストの設定に基づいて設定できます。 `VPP_DPDK_TX_QUEUES=4`

パラメータ	説明
VPP_DPDK_RX_DESCS=*value*	特定の設定がないすべてのインターフェイスの RX 記述子の数を指定します。有効な値の範囲は、インターフェイスのタイプとホストの設定に応じて 128 ～ 128,000 になります。デフォルト値は、指定されていないドライバ依存の値です。 次の例を使用して、RX記述子のデフォルトの数を 256 に設定します。 （注） 一致しない値は、インターフェイスのタイプとホストの設定に基づいて設定できます。 `VPP_DPDK_RX_DESCS=256`
VPP_DPDK_TX_DESCS=*value*	特定の設定がないすべてのインターフェイスの TX 記述子の数を指定します。有効な値の範囲は、インターフェイスのタイプとホストの設定に応じて 128 ～ 128,000 になります。デフォルト値は、VPP_DPDK_RX_DESCS または指定されていないドライバ依存の値です。 次の例を使用して、デフォルトの TX 記述子の数を 512 に設定します。 （注） 一致しない値は、インターフェイスのタイプとホストの設定に基づいて設定できます。 `VPP_DPDK_TX_DESCS=512`

インターフェイス固有の設定

表に示した次のパラメータは、個々のインターフェイスを調整します。これらのパラメータは、デフォルトの DPDK 設定（該当する場合）もオーバーライドします。

表 *7:* インターフェイスのパラメータ

パラメータ	説明
<ROLE>_VPP_RX_QUEUES=*value*	インターフェイスロールの RX キューの数を指定します。有効な値の範囲は、インターフェイスのタイプとホストの設定に応じて 1 ～ 64 になります。デフォルト値は未指定です。 サービスポート 1 ～ 2 の RX キューの数を設定するには、次の例を使用します。 `SERVICE1_INTERFACE_VPP_RX_QUEUES=2`

パラメータ	説明
<ROLE>_VPP_TX_QUEUES=*value*	インターフェイスロールの TX キューの数を指定します。有効な値の範囲は、インターフェイスのタイプとホストの設定に応じて 1 ～ 64 になります。デフォルト値は <ROLE>_VPP_RX_QUEUES、または unspecified です。
<ROLE>_VPP_RX_DESCS=*value*	インターフェイスロールの RX 記述子の数を指定します。有効な値の範囲は、インターフェイスのタイプとホストの設定に応じて 128 ～ 128,000 になります。デフォルト値は未指定です。 サービスポート 1 の RX 記述子の数を 1,024 に設定するには、次の例を使用します。 `SERVICE1_INTERFACE_VPP_RX_DESCS=1024`
<ROLE>_VPP_TX_DESCS=*value*	インターフェイスロールの RX 記述子の数を指定します。有効な値の範囲は、インターフェイスのタイプとホストの設定に応じて 128 ～ 128,000 になります。デフォルト値は <ROLE>_VPP_RX_DESCS、または unspecified です。
<ROLE>_VPP_WORKER_LIST=*value*	インターフェイスロールのワーカースレッドを指定します。ワーカーリストは、Linux プロセッサ番号をカンマで区切ったリストです。有効な値は 0 ～（CPU の数から 1 を差し引いた数）です。デフォルトは、すべてのソケットに割り当てられるラウンドロビン番号であり、すべてのソケットの最初のコアと最初のソケットの 2 番目のコアをスキップします。 サービスポート 1 のワーカースレッドリストを Linux プロセッサ番号 2 および 3 に設定するには、次の例を使用します。 `SERVICE1_INTERFACE_VPP_WORKER_LIST=2,3`

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-SI が正しくインストールされ、市販（COTS）サーバー上の仮想マシン（VM）上で実行されていることを前提としています。

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-SI 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。

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。