



仮想ネットワーク機能記述子について

- [仮想ネットワーク機能記述子の概要 \(1 ページ\)](#)
- [仮想ネットワーク機能記述子への拡張定義 \(1 ページ\)](#)

仮想ネットワーク機能記述子の概要

ESC は、VNF 特性を記述する TOSCA ベースの仮想ネットワーク機能記述子 (VNFD) をサポートします。VNFD は *GS NFV-SOL 001 v.2.7.1* の仕様および ETSI によって指定された標準規格に準拠しています (YAML バージョン 1.2 では *TOSCA Simple Profile* を実装しています)。

VNFD ファイルには、VNF のインスタンス化パラメータと操作の動作が記述されています。これには、KPI、およびオンボーディングと VNF のライフサイクル管理のプロセスで使用できるその他の主要な要件が含まれています。

VNF ライフサイクル操作については、[VNF ライフサイクル操作](#)を参照してください。

仮想ネットワーク機能記述子への拡張定義

ESC はシスコが定義した VNFD の拡張機能を実装しており、ESC ではサポートされているものの ETSI 標準で明確に定義されていない、より高度な概念を公開しています。これらの拡張機能は、他の ETSI MANO コンポーネントとの最大限の互換性を確保するために、ETSI に準拠した方法で実装されています。

展開ごとにこれらのプロパティを制御する必要がある場合は、着信要求で `additionalParams` として指定できる VNFD の入力で、ハードコードされた値を置き換えます。

VNFCs (`tosca.nodes.nfv.Vdu.Compute`)

コンピューティングノードでは、拡張

`tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties` を介して多くの ESC 機能を公開できます。これには、次の事項が含まれます。

- VIM 上の VNFC の自動生成された名前のオーバーライド。
- VIM フレーバ (VNFC に指定された ETSI 機能をオーバーライド)。

- このタイマーが期限切れになるまでアクションが実行されるのを防ぐため、ESCに予想されるブートアップ時間を指定。
- VNFC を展開後、実行/保存する Day-0 設定ブロックを提供。
- モニタリングエージェントを設定するための KPI パラメータと関連ルールの指定。
- VM グループ内の配置ルール。

データ型拡張の定義を次に示します。

```
data_types:
...
cisco.datatypes.nfv.VnfcAdditionalConfigurableProperties:
  derived_from: tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties
  properties:
    vim_flavor:
      type: string
      required: true
    bootup_time:
      type: integer
      required: true
    vm_name_override:
      type: string
      required: false
    recovery_action:
      type: string
      required: true
    recovery_wait_time:
      type: integer
      required: true
    monitor_on_error:
      type: boolean
      description: Continue monitoring of VNFC on error state.
      required: false
    max_retries:
      type: integer
      description: The number of recovery attempts
      required: false
    kpi_data:
      type: map # key: event_name
      description: The different KPIs applicable to this VDU
      required: false
      entry_schema:
        type: cisco.datatypes.nfv.data.Kpi
        description: A single KPI
    admin_rules:
      type: map # key: event_name
      description: Actions for events
      required: false
      entry_schema:
        type: cisco.datatypes.nfv.data.Admin_rules
        description: Define actions for events
    name_override:
      type: string
      description: An optional custom name that be be configured on the VIM
      required: false
    vendor_section:
      type: cisco.datatypes.nfv.VendorExtension
      required: false

cisco.datatypes.nfv.VnfcConfigurableProperties:
  derived_from: tosca.datatypes.nfv.VnfcConfigurableProperties
```

```

properties:
  additional_vnfc_configurable_properties:
    type: cisco.datatypes.nfv.VnfcAdditionalConfigurableProperties
    required: false

node_types:
  cisco.nodes.nfv.Vdu.Compute:
    derived_from: tosca.nodes.nfv.Vdu.Compute
    properties:
      configurable_properties:
        type: cisco.datatypes.nfv.VnfcConfigurableProperties
        description: Describes the configurable properties of all VNFC instances based
on this VDU
        required: false

```

次に例を示します。

```

vdu1:
  type: tosca.nodes.nfv.Vdu.Compute
  properties:
    name: Example VDU1
    description: Example VDU
    boot_order:
      - boot1-volume
    configurable_properties:
      additional_vnfc_configurable_properties:
        vim_flavor: Automation-Cirros-Flavor
        bootup_time: 1800
        vm_name_override: my-vdu-1
        recovery_action: REBOOT_THEN_REDEPLOY
        recovery_wait_time: 100
        monitor_on_error: false
        max_retries: 2
      kpi_data:
        VM_ALIVE-1:
          event_name: 'VM_ALIVE-1'
          metric_value: 1
          metric_cond: 'GT'
          metric_type: 'UINT32'
          metric_occurrences_true: 1
          metric_occurrences_false: 30
          metric_collector:
            type: 'ICMPPing'
            nicid: 1
            poll_frequency: 10
            polling_unit: 'seconds'
            continuous_alarm: false
    admin_rules:
      VM_ALIVE-1:
        event_name: 'VM_ALIVE-1'
        action:
          - 'ALWAYS log'
          - 'FALSE recover autohealing'
          - 'TRUE esc_vm_alive_notification'
    placement_type: zone
    placement_target: nova
    placement_enforcement: strict
    vendor_section:
      cisco_esc:
        config_data:
          example.txt:
            file: ../Files/Scripts/example.txt
            variables:
              DOMAIN_NAME: { get_input: DOMAIN_NAME }
              NAME_SERVER: { get_input: NAME_SERVER }

```

```

VIP_ADDR: { get_input: VIP_ADDR }
VIP_PREFIX: { get_input: VIP_PREFIX }
vdu_profile:
  min_number_of_instances: 1
  max_number_of_instances: 1
  capabilities:
virtual_compute:
  properties:
    virtual_cpu:
      num_virtual_cpu: 8
    virtual_memory:
      virtual_mem_size: 16
requirements:
  - virtual_storage: cdr1-volume
  - virtual_storage: boot1-volume

```

`vm_name_override` が指定されていない場合、ESC によって VM 名が自動生成されます。

ESCは、VNFCを表すコンピューティングノードに与えられたラベルと一致する `vduId` によって識別される VNFC の `VnfInstance.instantiatedVnfInfo.vnfcResourceInfo.metadata.vim_vm_name` に VNFC 固有の値を保存します。



(注) 多数の入力パラメータを指定できるため、複数の展開で1つのテンプレートを 사용할 ことができます。

接続ポイント (cisco.nodes.nfv.VduCp)

VduCp ノードタイプに対するシスコの拡張機能では、主にインターフェイス要件マップを定義 できます。接続ポイントに追加された機能は次のとおりです。

- VIM 上のポートの自動生成された名前のオーバーライド
- ポートが管理ポート (モニタリングに使用する) かどうかの識別
- 許可済みアドレスペア¹
- 特定のネットワークカードタイプとインターフェイスタイプ (SR-IOV など) のサポート
- ポート バインディング プロファイルのサポート
- ポートセキュリティが有効かどうか

次に例を示します。

```

vdu1_nic0:
  type: cisco.nodes.nfv.VduCp
  properties:
    layer_protocols: [ ipv6 ]
    protocol:
      - associated_layer_protocol: ipv6
    trunk_mode: false
    order: 0
    allowed_address_pairs:
      - ip_address: { get_input: VDU1_NIC0_AADR_PAIRS }
  virtual_network_interface_requirements:
    - support_mandatory: true
      network_interface_requirements:

```

¹ 複雑すぎてマップに含めることができないため、仕様外の拡張です

```

nw_card_model: virtio
iface_type: direct
management: true
name_override: my-vdu1-nic0

port_security_enabled: false
binding_profile:
  trusted: true
requirements:
  - virtual_binding: vdu1

```

ESC ETSI NFV MANO は、Cisco ネットワーク要件を使用する SR-IOV プロパティをサポートします。上の例のようにタイプを `direct` に指定することで、VNFC を SR-IOV パススルーアダプタに関連付けるよう、インターフェイスを設定できます。

展開ごとにこれらのプロパティを制御する必要がある場合は、着信要求で `additionalParams` として指定できる VNFD の入力で、ハードコードされた値を置き換えます。



(注) ポートバインディングプロファイルは、OpenStack の Pike 以降のバージョンで使用できます。

ボリューム (`tosca.nodes.nfv.Vdu.VirtualBlockStorage`)

ESC は、シスコの拡張としてアウトオブバンドボリュームをサポートします。これにより、永続的なボリューム UUID の仕様を、`VirtualBlockStorage` ノードに対する `resourceId` プロパティとして、VNFD で定義されたエフェメラルボリュームの代わりに使用できます。余分なプロパティを追加する代わりに、ESC は VIM からの UUID で識別することによって、VNFD で指定されたボリュームをオーバーライドし、独自の永続的な（展開されたアウトオブバンド）ストレージを指定することを許可します。

次に例を示します。

```

boot1-volume:
  type: toasca.nodes.nfv.Vdu.VirtualBlockStorage
  properties:
    virtual_block_storage_data:
      size_of_storage: 4GB
      vdu_storage_requirements:
        resource_id: { get_input: VDU1_BOOT_VOL_UUID }
        vol_id: 1
        bus: ide
        type: LUKS
    sw_image_data:
      name: 'Automation_Cirros'
      version: '1.0'
      checksum: 9af30fce37a4c5c831e095745744d6d2
      container_format: bare
      disk_format: qcow2
      min_disk: 2 GB
      size: 2 GB
  artifacts:
    sw_image:
      type: toasca.artifacts.nfv.SwImage
      file: ../Files/Images/Automation-Cirros.qcow2

```



- (注) VNFD は、MiB、GiB、TiB 相当などのメビバイト単位のボリュームまたはソフトウェアイメージサイズを受け入れます。ボリュームまたはソフトウェアイメージのサイズが MB、GB、TB などのメガバイト単位の場合、ESC はサイズをメビバイト単位に変換し、最も近い値に調整します。わかりやすくするために、ボリュームまたはソフトウェアイメージのサイズには、必ずメビバイト単位を使用してください。

セキュリティグループルール (cisco.nodes.nfv.VduCp)

上記のボリュームの永続的な処理に従って、ESC は VNFD で設定する代わりに、アウトオブバンドセキュリティグループを指定する機能を提供します。これは、標準のドキュメントでセキュリティグループを説明するために使用される動詞が、非常に複雑な設定に対しては単純すぎるためです。セキュリティグループは接続ポイントで使用するために指定されるため、VNFD で定義されます。

次に例を示します。

```
c1_nic0:
  type: cisco.nodes.nfv.VduCp
  properties:
    order: 0
    layer_protocols: [ ipv6 ]
    protocol:
      - associated_layer_protocol: ipv6
    trunk_mode: false
    virtual_network_interface_requirements:
      - support_mandatory: true
        network_interface_requirements:
          management: "false"
          name_override: { get_input: C1_SRIOV_A_INT_NAME }
          iface_type: "direct"
  metadata:
    security_groups: { get_input: VIM_NETWORK_SEC_GRP_0 }
  requirements:
    - virtual_binding: c1
```

仮想リンク (tosca.nodes.nfv.VnfVirtualLink)

VNFD で定義された仮想リンクを使用して、これらの物理プロバイダーネットワークを定義できます。

VNFD の例 :

```
vpc-di-internall1:
  type: tosca.nodes.nfv.VnfVirtualLink
  properties:
    connectivity_type:
      layer_protocols: [ ipv4 ]
    description: DI Internal 1 Network VL
    vl_profile:
      max_bitrate_requirements:
        root: 100000
      min_bitrate_requirements:
        root: 0
    virtual_link_protocol_data:
      - associated_layer_protocol: ethernet
        l2_protocol_data:
```

```
network_type: vlan
segmentation_id: { get_input: VL1_SEG_ID }
physical_network: vlan_network
```

また、DHCP を使用してアドレスを割り当てるときに内部接続ポイントが使用できる IP サブネットワークを指定するためにも使用できます。

VNFD の例 :

```
vpc-di-internal2:
  type: toasca.nodes.nfv.VnfVirtualLink
  properties:
    connectivity_type:
      layer_protocols: [ ipv4 ]
    description: DI Internal 1 Network VL
    vl_profile:
      max_bitrate_requirements:
        root: 100000
      min_bitrate_requirements:
        root: 0
    virtual_link_protocol_data:
      - associated_layer_protocol: ipv4
        l3_protocol_data:
          ip_version: ipv4
          cidr: 1.180.10.0/29
          dhcp_enabled: true
```

ライフサイクル管理操作の詳細については、「[VNF ライフサイクルの管理](#)」を参照してください。



(注) ESC の以前のバージョンは、上記の機能をサポートするためにシスコ専用の拡張機能をサポートしていました。これらの拡張機能は仕様の範囲外でした。また、現在これらの拡張機能は SOL001 標準にほとんど準拠していますが、下位互換性のために以前の定義は引き続き ESC でサポートされています。詳細については、Cisco Elastic Services Controller 5.4 マニュアル *vm_name* を参照してください。

