仮想ネットワーク機能の修復

ETSI API を使用した仮想ネットワーク機能の修復

ESC は、ライフサイクル管理の一環として、障害が発生すると VNF を修復します。展開中に指定したリカバリポリシーがリカバリを制御します。ESCは、ポリシー主導型のフレームワークを使用したリカバリをサポートしています。詳細については、『Cisco Elastic Services Controller User Guide』の「Configuring a Recovery Policy Using the Policy-driven Framework」を参照してください。

修復パラメータは、VNF を修復する通知をトリガーするためにモニタする動作を定義します。これらのパラメータは、ルールを使用して VNFD の各コンピューティングノードの KPI セクションで構成されます。ルールは、これらの KPI 条件の結果として、VNF を修復するアクションを定義します。

ETSI VNFM は、次の 2 つのセクションを使用してモニタリングを設定します。

- kpi_data：モニタリングのタイプ、イベント、ポーリング間隔、およびその他のパラメータを定義します。
- admin_rules：KPI モニタリングイベントがトリガーされたときのアクションを定義します。

例：

```
vdu1:
  type: cisco.nodes.nfv.Vdu.Compute
  properties:
    name: Example VDU1
    description: Example VDU
    ...
     configurable_properties:
      additional_vnfc_configurable_properties:
        vim_flavor: { get_input: VIM_FLAVOR }
        bootup_time: { get_input: BOOTUP_TIME }
```

```
vm_name_override: { get_input: VDU1_VM_NAME}
recovery_action: REBOOT_THEN_REDEPLOY
recovery_wait_time: 1
kpi_data:
  VM_ALIVE-1:
    event_name: 'VM_ALIVE'
    metric_value: 1
    metric_cond: 'GT'
    metric_type: 'UINT32'
    metric_occurrences_true: 1
    metric_occurrences_false: 30
    metric_collector:
      type: 'ICMPPing'
      nicid: 1
      address_id: 0
      poll_frequency: 10
      polling_unit: 'seconds'
      continuous_alarm: false
admin_rules:
  VM_ALIVE:
    event_name: 'VM_ALIVE'
    action:
      - 'ALWAYS log'
      - 'FALSE recover autohealing'
      - 'TRUE esc_vm_alive_notification'
```

前の例は、デフォルトのKPIと、ESCでの展開を完了するために必要なサービスアライブ通知をサポートするルールを示しています。VNFD で公開される KPI、ルール、および基盤となるデータモデルの詳細については、『Cisco Elastic Services Controller User Guide』の「KPIs、Rules and Metrics」を参照してください。

VNFのリカバリは、初期導入時またはリカバリ要求で定義されたリカバリポリシーによって決定された、影響を受ける VNFC に対するアクションを要求することです。

リカバリには4種類のアクションがあります。インスタンスに注意が必要であることを示すイベントが受信されると、タイマーが期限切れになるか、手動のリカバリ要求が受信されます。修復ワークフローは、デフォルトで、VNF レベルまたは VNFD 内の VNFC レベルで設定されたリカバリポリシーを使用します。サポートされているポリシーは次の通りです。

- REBOOT_THEN_REDEPLOY：最初に、影響を受けたVNFCの再起動を試みます。これが失敗した場合、影響を受けた VNFC の再展開（同じホスト上で）を試みます
- REBOOT_ONLY：VM の再起動のみを試みます
- RESET_THEN_REBOOT — VM の状態をリセットして (Openstack のみ)、VM の再起動を試みます。
- REDEPLOY_ONLY：VM の再展開のみを試みます

リカバリポリシーが VNF レベルで設定されている場合、ポリシーは各構成要素 VNFC に適用されます。VNFCレベルで指定されている場合は、そのポリシーが優先されます。モニタリングエージェントが各VNFCをモニタし、リカバリ状況になると、メッセージがアラームに変換され、登録されたコンシューマ（NFVO または Element Manager）に送信されます。

HealVnfRequest には、リカバリ要求の処理中に VNFM 内でさまざまな動作をトリガーする原因パラメータが含まれています。原因がVNFMでサポートされている値の1つである場合（およびサポートされている原因として展開のVNFDにリストされている場合）、次の表に示すよ

うに、特定の追加の *Params* キーがアクティブ化されて、必要なリカバリアクションをサポートします。NFVO が原因をサポートしている場合、許可は *additionalParams* を受け取り、リカバリ要求を実行する前に入力を変更できるようにします。

原因が ESC でサポートされているオーバーライドの原因の 1 つでない場合、提供された値は単なるメタデータであると見なされ、無視されます。 VNFM は、展開時に構成されたリカバリポリシーを使用します。原因が ESC によってサポートされていても、VNFD にリストされていない場合、要求は拒否されます。

表 *1 : HealVnfRequest* の原因

原因	additionalParams キー	リカバリ動作

APPLICATION_FAILURE	オプション
	vnfcInstanceId
		これらの VNFC のみにリカバリを制限する VNFC インスタンスの有効な識別子のリストが vnfcInstanceId に入力されていない限り、リカバリは VNF 全体を再起動しようとします。次に例を示します。
		{
		...
		"vnfcInstanceId": [
		"resId1",
		"resId2"
		]
		...
		}

原因	additionalParams キー	リカバリ動作
VIRTUALISATION_FAILURE	オプション vnfcInstanceId resourceId virtualStorageDescId	vnfcInstanceId は、APPLICATION_FAILURE ごとに処理されます。 さらに、同じ要求で置換される永続ボリュームがある場合、複数の要求を回避するために、VNFD および VIM 内のボリュームの識別子が提供されます。ただし、ボリュームが接続されている VNFC においては、修復される VNFC のリストに含まれている必要があります。**この永続ボリュームの更新は、Openstack VIM にのみ適用されます。** 障害があるかまたは削除された VNFM によって管理される一時的なポートおよびボリュームは、再作成されて接続され、リカバリが確実に成功するようにします。
APPLICATION_OR_VIRTUALISATION_FAILURE	オプション vnfcInstanceId	APPLICATION_FAILURE ごとに処理されます。 障害があるか削除された VNFM によって管理されるエフェメラルポートおよびボリュームは再作成され、接続され、VM が再展開された場合にリカバリが成功するようにします。
INVALID_VM_STATE	オプション vnfcInstanceId	APPLICATION_FAILURE ごとに処理されます。

原因	additionalParams キー	リカバリ動作
PERSISTENT_VOLUME_FAILURE	必須： resourceId virtualStorageDescId オプション vnfcInstanceId	vnfcInstanceId は、APPLICATION_FAILURE ごとに処理されます。必須キーを使用すると、VM を再展開することなく、新しい永続ボリュームで既存のボリュームを置き換えることができます。データモデルが更新され、ボリュームが置き換えられたら、VM が再起動されます。**これは、Openstack VIM にのみ適用されます。**
CHANGE_PERSISTENT_VOLUME	必須： resourceId virtualStorageDescId	必須キーを使用すると、VM を再展開することなく、新しい永続ボリューム（multi-attach を含む）で既存のボリュームを置き換えることができます。データモデルが更新され、ボリュームが置き換えられると、VM が再起動されます。**これは、Openstack VIM にのみ適用されます。**
VIM_FAILURE	なし	additionalParams キーはアクティブ化されませんが、NFVO からの許可には、使用可能な VIM に VNF を再展開するための新しい vimConnectionInfo が含まれている**必要**があります。そうしない場合、リカバリ要求は拒否されます。 （注） この原因が使用された場合、VIM は使用できないと想定されるため、古い展開は削除されません。 VIM が再び到達可能になったら、手動で削除する必要があります。

VNF インスタンスで自動修復が有効になっている場合、ESC は展開時に設定されたリカバリポリシーに基づいて VNF のリカバリを自動的に試みます。これは、VNFD で構成するか、インスタンス化の前に VNF インスタンスに対して変更することができます。

自動修復フラグ（*isAutohealEnable*）VNF インスタンスリソースを変更するには、仮想ネットワーク機能の変更を参照してください。

自動修復が有効でない場合、アラームのみがすべてのサブスクライバにディスパッチされます。サブスクライバーは、次の例に従って、手動の HealVnfRequest を開始できます。パラメータはデフォルトではオプションですが、さまざまな原因について表 9 のルールが適用されます。

SOL003 の例：

メソッドタイプ：

```
POST
```

VNFM エンドポイント：

```
/vnf_instances/{vnfInstanceId}/heal
```

HTTP 要求ヘッダー：

```
Content-Type:application/json
```

要求ペイロード（ETSI データ構造：HealVnfRequest）

```
{
"cause":"VIRTUALISATION_FAILURE",
"additionalParams": {
 "virtualStorageDescId": "cf-cdr1-vol",
 "resourceId": " d8771acb-a32f-66dg-7bc2-8f4ec333ccb8"
},
"vnfcInstanceId": [b9909dde-e21e-45ec-9cc0-9e9ae413eee0"]
}
```

SOL002 の例：

```
POST /vnf_instance/{vnfInstanceId}/heal
 {
   "vnfcInstanceId": ["b9909dde-e21e-45ec-9cc0-9e9ae413eee0"],
   "cause": "b9909dde-e21e-45ec-9cc0-9e9ae413eee0"
 }
```

`vnfcInstanceIds` のリストは、リカバリを必要な VNFC に制限します。ただし、このリストがないということは、要求が VNF 全体に適用されることを意味します。

SOL002 HealVnfRequest の原因は、SOL003 API と同じ動作をします。

モニタリングの詳細については、ETSI API を使用した仮想ネットワーク機能のモニタリングを参照してください

修復中の VM の回復

リカバリアクションが REDEPLOY_ONLY または REBOOT_THEN_REDEPLOY であり、SOL002 および SOL003 の修復操作中に VM を再展開する必要がある場合は、次のことを確認します。

- エフェメラルボリュームがないか、またはエラー状態です。それらを再作成します。
- エフェメラル neutron ポートがないか、またはエラー状態です。 それらを再作成します。

（注） vnfcInstanceIds が修復ペイロードで提供されている場合、SOL002 修復は特定の VNFC に制限されます。

修復中の既存の展開の更新

展開が正常に作成されたら、その中のリソースを更新できます。展開管理の一環として、リソースを追加または削除したり、既存のリソースの設定を更新したりできます。これらの更新は、実行中の展開で実行できます。リソースは、リカバリプロセスの一環として更新されます。

修復ワークフロー中に、（ETSI NFV MANO API を介してプロビジョニングされた）既存の展開を更新できます。修復要求中に、既存のイメージと Day-0 パラメータが比較され、後続の修復要求の一部として提供される新しいパラメータに更新されます。

ヒーリングワークフローでは、次のことが可能です。

- 展開モデルを新しいイメージと Day-0 設定で更新する
- アップグレードされたイメージによる修復時に、新規または既存の設定データを VNFC に再適用する

（注） 変更が VIM で直接実行されない場合、データモデルの更新後に VNF を再展開する必要があります。

HealVnfRequest を介して新しい *additionalParams* を指定した後、（NFVO からの）付与応答も新しいイメージまたは新しい *additionalParams* を指定する場合、これもサービス更新をトリガーします。

展開を再展開の一環として移動させる必要があると NFVO が判断した場合、付与はリソースの新しい配置を反映するための新しい *zoneId* を提供します。

リカバリアクションは、サービスの更新が完了した後に実行されます。再展開の場合は、最新の展開モデルを考慮して、展開された更新が元に戻されないようにします。

次の例は、新しい *additionalParams* や新しい *vimSoftwareImageId* でサービス更新をトリガーするために、NFVO が付与に返す詳細を示しています。

例：

```
{
  "headers" : {
    "Content-Type" : [ "application/json" ],
    "Location" : [
"http://{nfvoApiRoot}/sol003/default/grant/v1/grants/38ba2103-dab3-450e-992b-ee85aad6c899"
 ],
    "Content-Length" : [ "22935" ],
  },
  "body" : {
    "id" : "38ba2103-dab3-450e-992b-ee85aad6c899",
    "vnfInstanceId" : "6aaf527c-0093-49c3-ba2e-49fc6d8a4f71",
    "vnfLcmOpOccId" : "cdc5d9b3-81a0-400b-a4d9-97d1b3e117d9",
    "_links" : {
      "self" : {
        "href" :
"http://{nfvoApiRoot}/sol003default/grant/v1/grants/38ba2103-dab3-450e-992b-ee85aad6c899"

      },
      "vnfLcmOpOcc" : {
        "href" :
"https://{vnfmApiRoot}/vnflcm/v2/vnf_lcm_op_occs/cdc5d9b3-81a0-400b-a4d9-97d1b3e117d9"
      },
      "vnfInstance" : {
        "href" :
"https://{vnfmApiRoot}/vnflcm/v2/vnf_instances/6aaf527c-0093-49c3-ba2e-49fc6d8a4f71"
      }
    },
    "vimConnections" : {
    "default_openstack_vim": {
      "vimType" : "OPENSTACK_V3",
      "vimId" : "595b0bc2-8dad-4087-abdf-ebe3b0b14d96",
      "interfaceInfo" : {
        "endpoint" : "https://{vimApiRoot}/v3"
      },
      "accessInfo" : {
        "password" : "********",
        "project" : "cisco",
        "projectDomain" : "demo",
        "region" : "RegionOne",
        "userDomain" : "demo",
        "username" : "********"
      }
    } },
    "zones" : [{
      "id" : "1773873a-ab15-4a7b-b024-bc338425ed24",
      "zoneId" : "nova"
    },{
      "id" : "1773873a-ab15-4a7b-b024-bc555555ed55",
      "zoneId" : "nova2"
    }],
    "addResources" : [{
      "resourceDefinitionId" : "res-a6252dbf-b418-4f88-b8a9-14d8f3942938",
      "vimConnectionId" : "myVimConnection",
      "zoneId" : "1773873a-ab15-4a7b-b024-bc555555ed55"
    }],
    "vimAssets" : {
      "softwareImages" : [ {
        "vnfdSoftwareImageId" : "s3",
```

```
        "vimSoftwareImageId" : "3a609da7-e2b2-4e27-91b6-7bcabe902820",
        "vimConnectionId" : "myVimConnection"
      }, {
        "vnfdSoftwareImageId" : "s4",
        "vimSoftwareImageId" : "3a609da7-e2b2-4e27-91b6-7bcabe902820",
        "vimConnectionId" : "myVimConnection"
      } ]
    }
  },
  "additionalParams": [
    ...
    /* changed additionalParams */
    "CF_VIP_ADDR": "10.123.23.4",
    "SF_VIP_ADDR": "10.123.24.4",
    ...
  ],
  "statusCode" : "CREATED",
  "statusCodeValue" : 201
}
```

修復の詳細については、ETSI API を使用した仮想ネットワーク機能の修復 （1 ページ）を参照してください。

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。