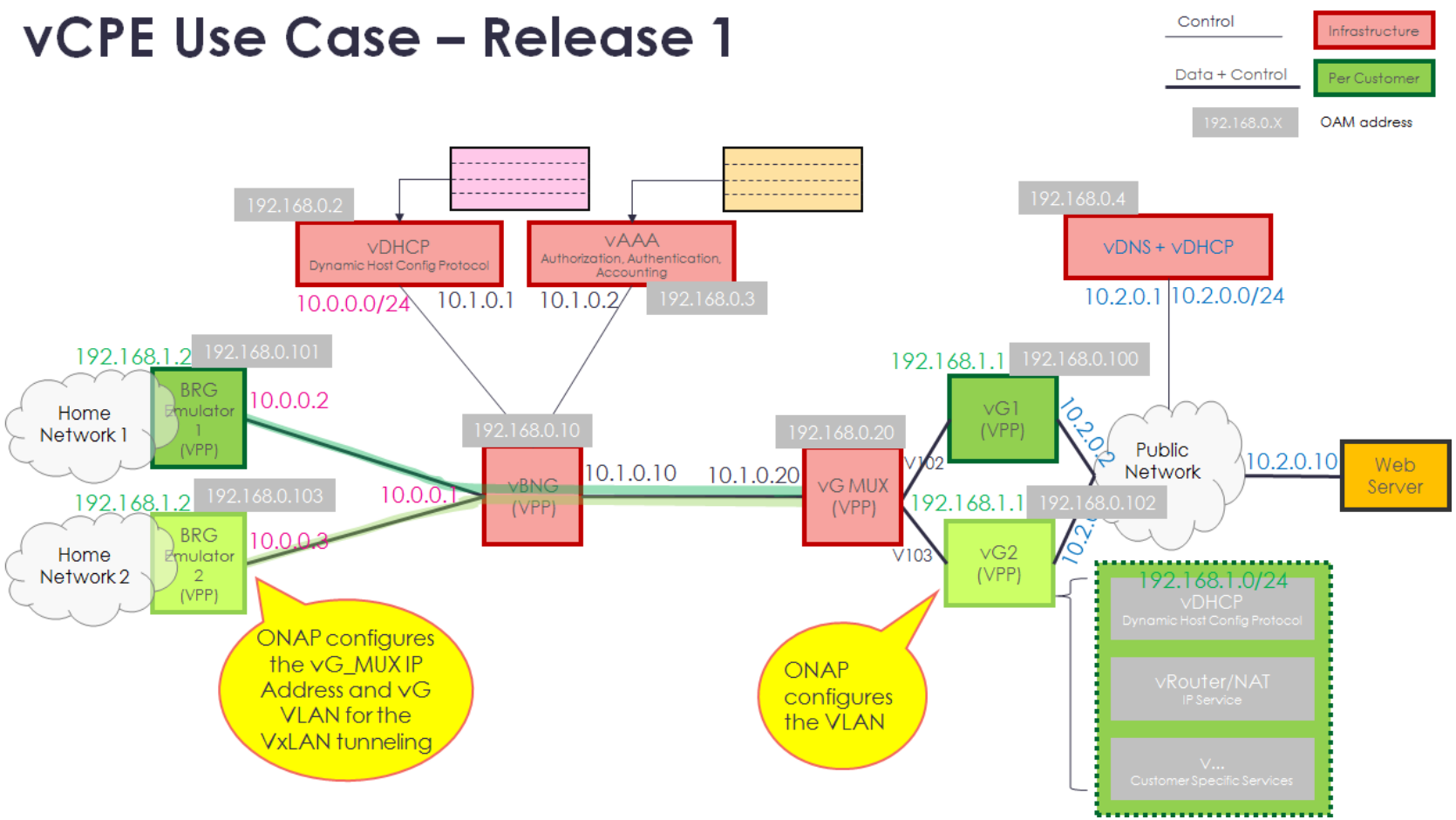
vCPE Use Case Review

Integration Project and Use Case Subcommittee

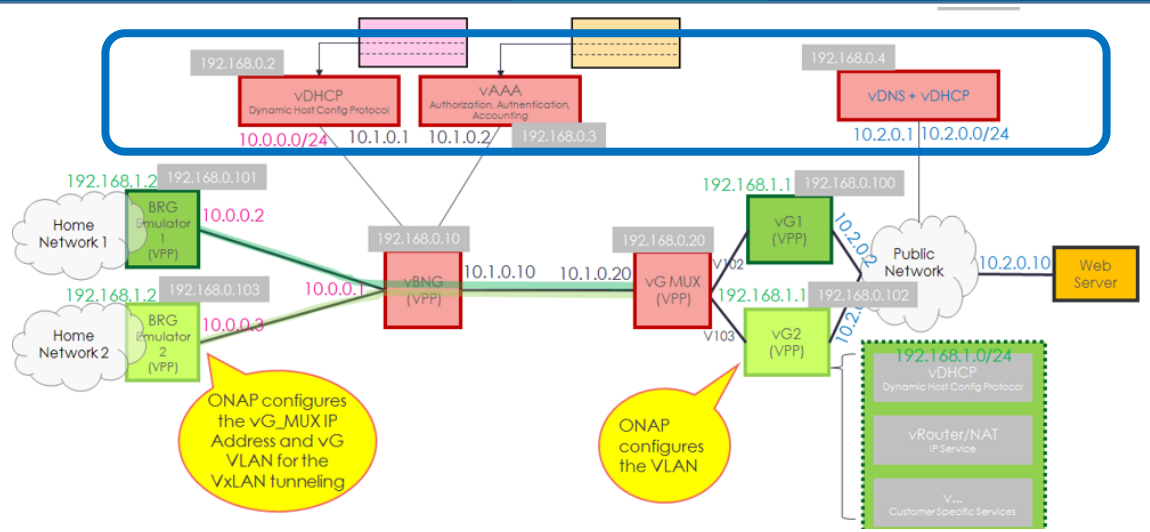
July 25 , 2017

vCPE Use Case – Release 1

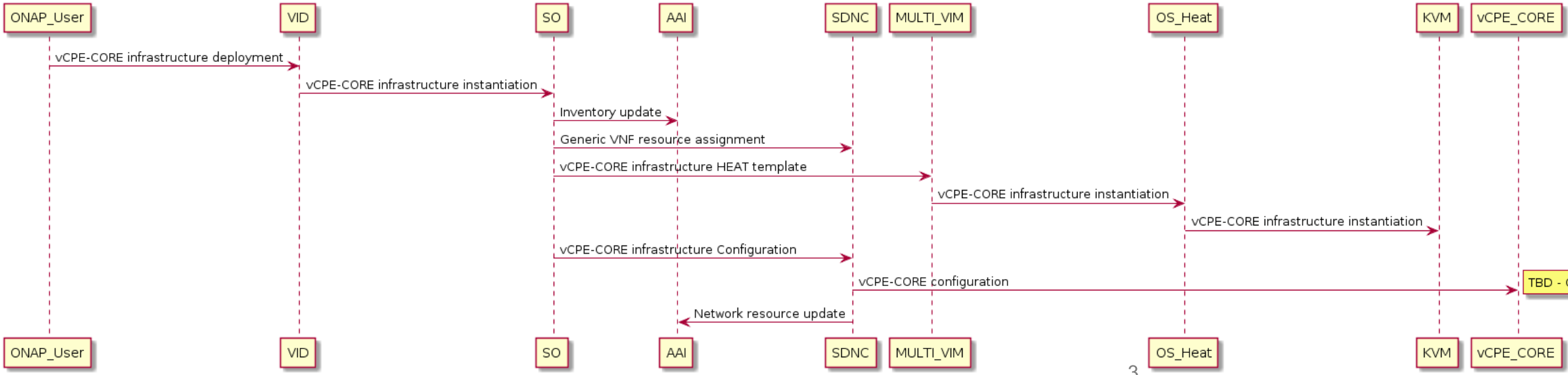


- Red blocks are infrastructure. They are shared among all the users.
- Green blocks are used by customers. Each customer needs a vBRG and a vG.
- Data plane: packet exchange between vBRG and Web Server.
- vBRG, vBNG, vGMUX, and vG are all based on VPP.
- vDHCP, vAAA, vDNS, and Web Server are all open source applications.

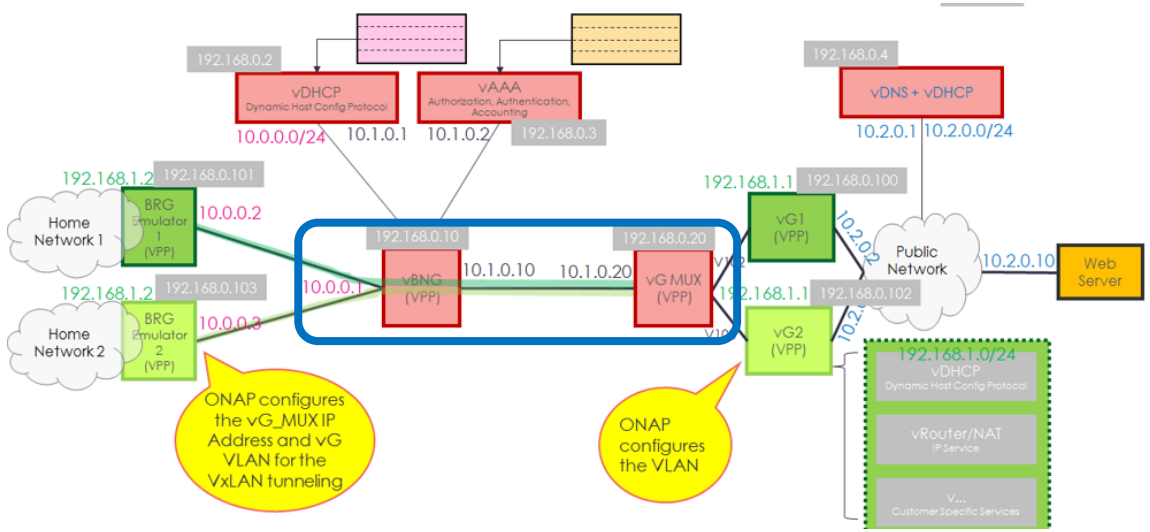
Infrastructure instantiation: vDHCP, vAAA, vDNS



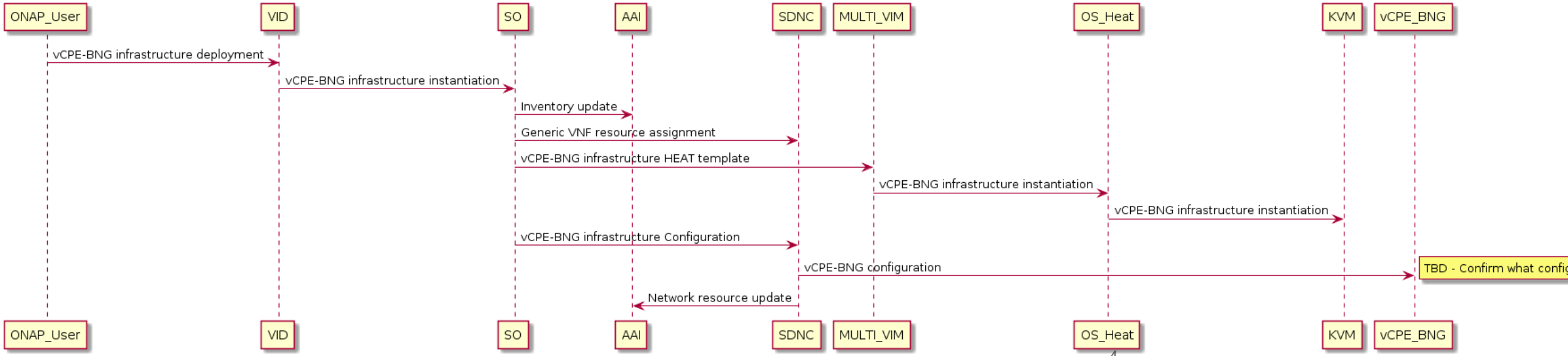
vCPE-CORE infrastructure instantiation



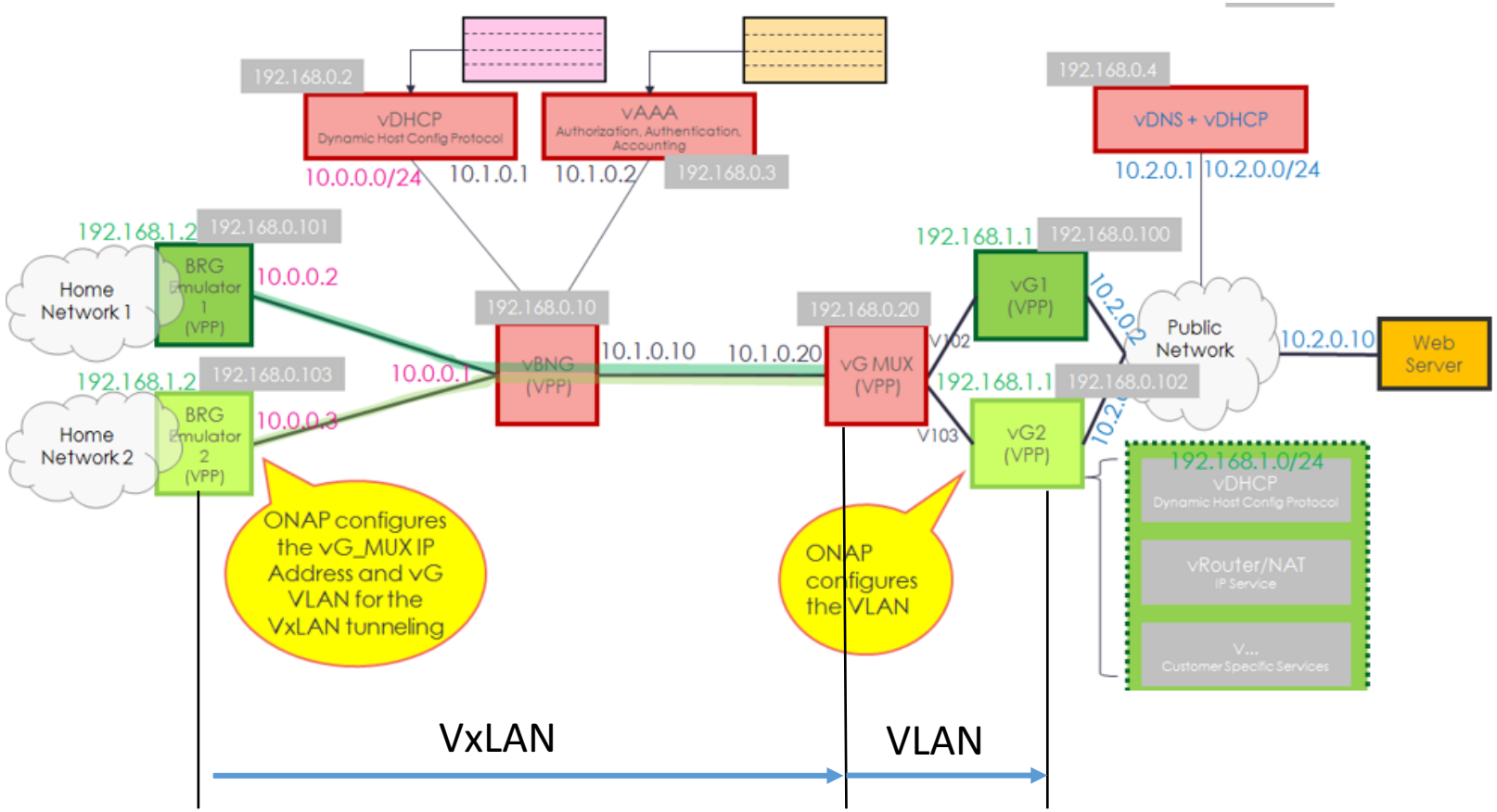
Infrastructure instantiation: vBNG, vGMUX



vCPE-BNG infrastructure instantiation



Data Plane: VxLAN and VLAN

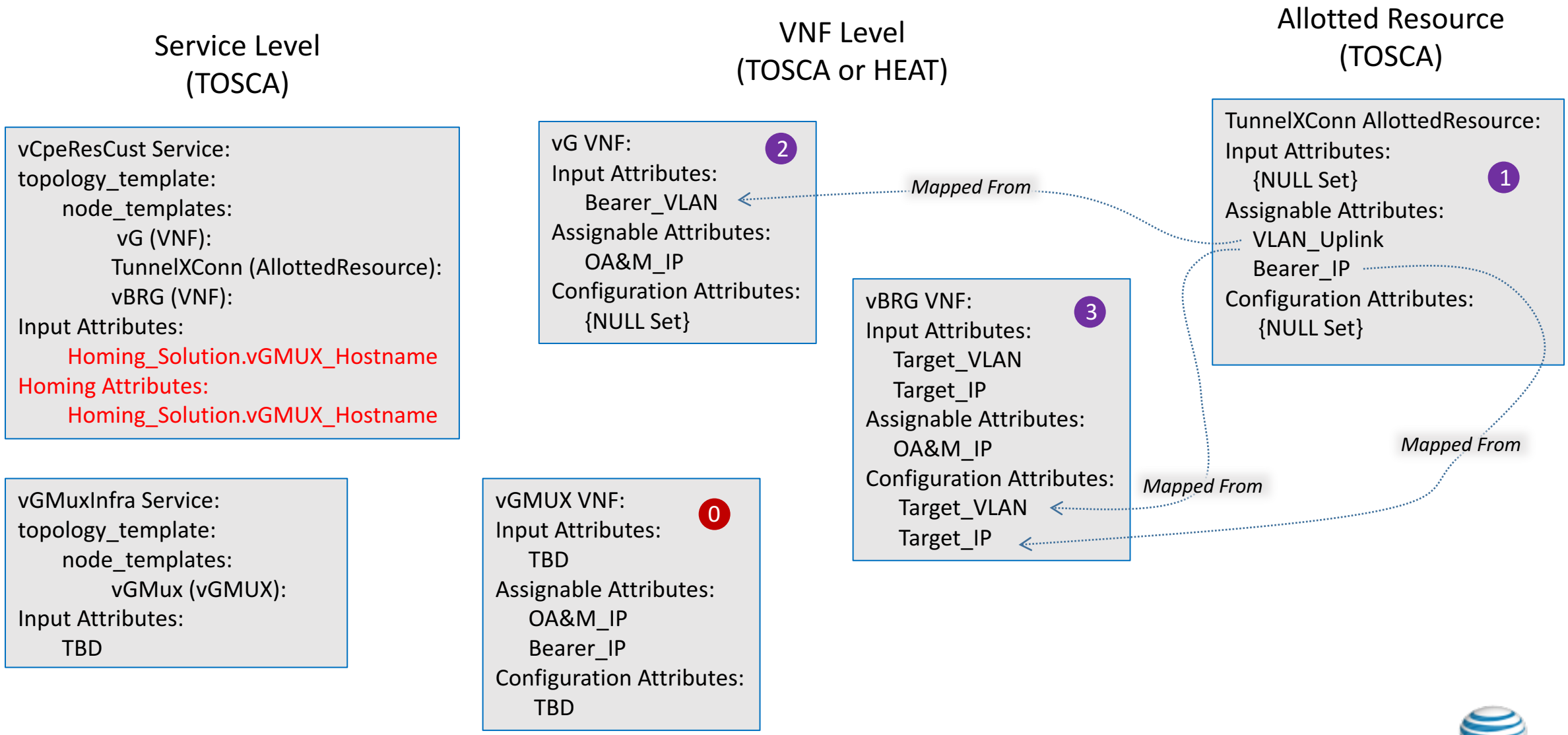


Packets from BRG carries:

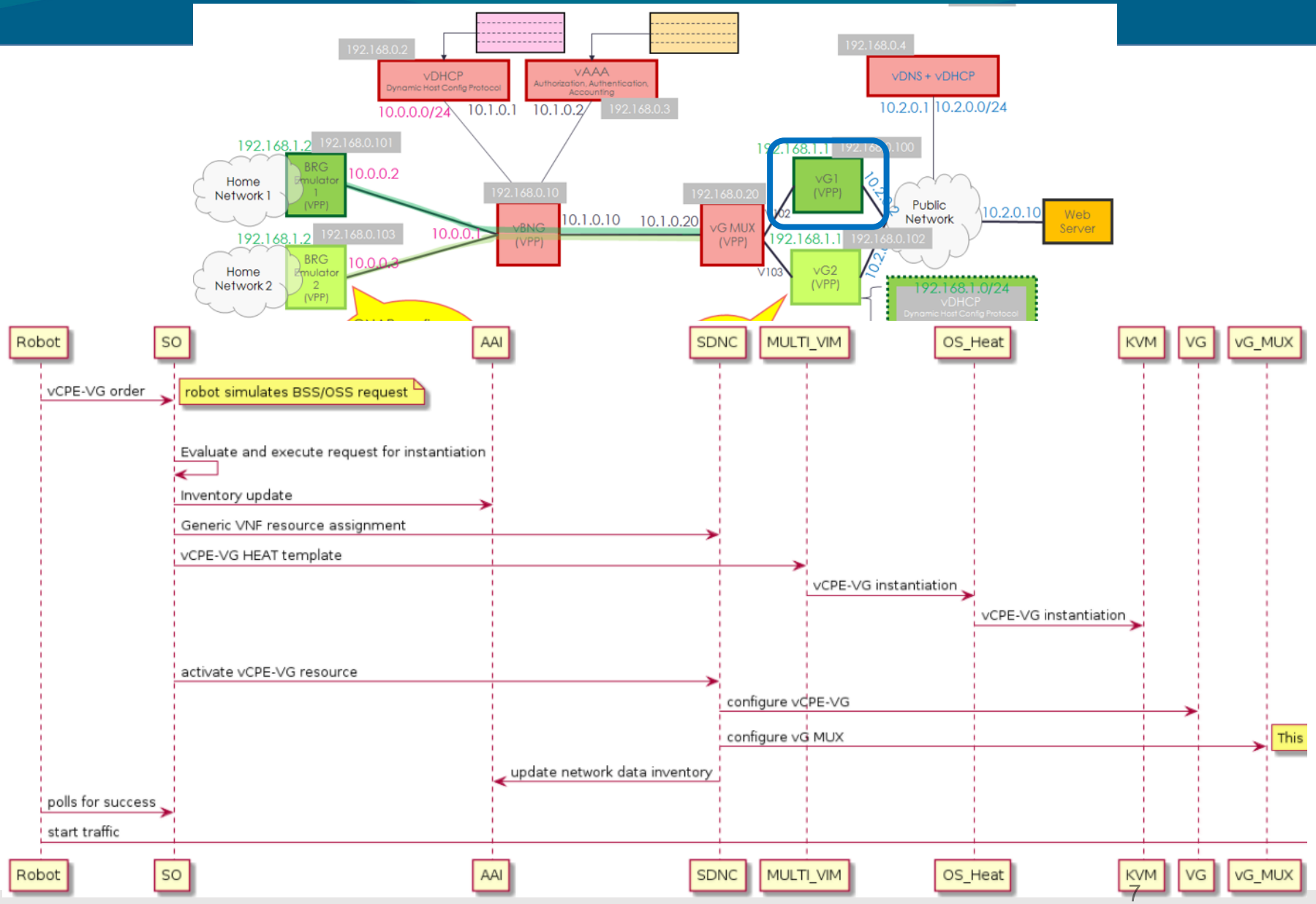
- vGMUX IP
- VxLAN ID
- VLAN ID

The info is used by vGMUX to fill lookup table and forward packet to the right vG

Residential Broadband vCPE Use Case Model: Data Mappings (Release 1)



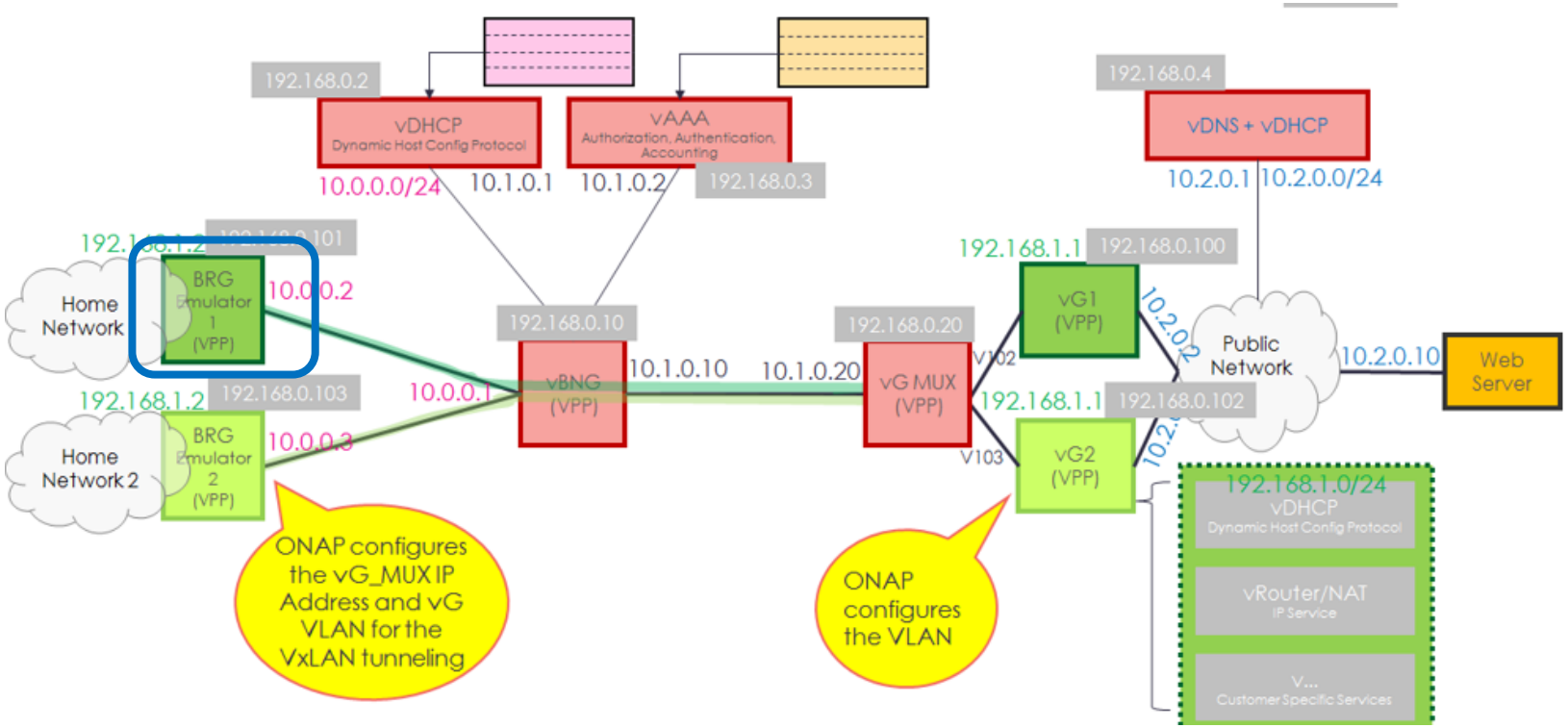
Customer service instantiation: vG

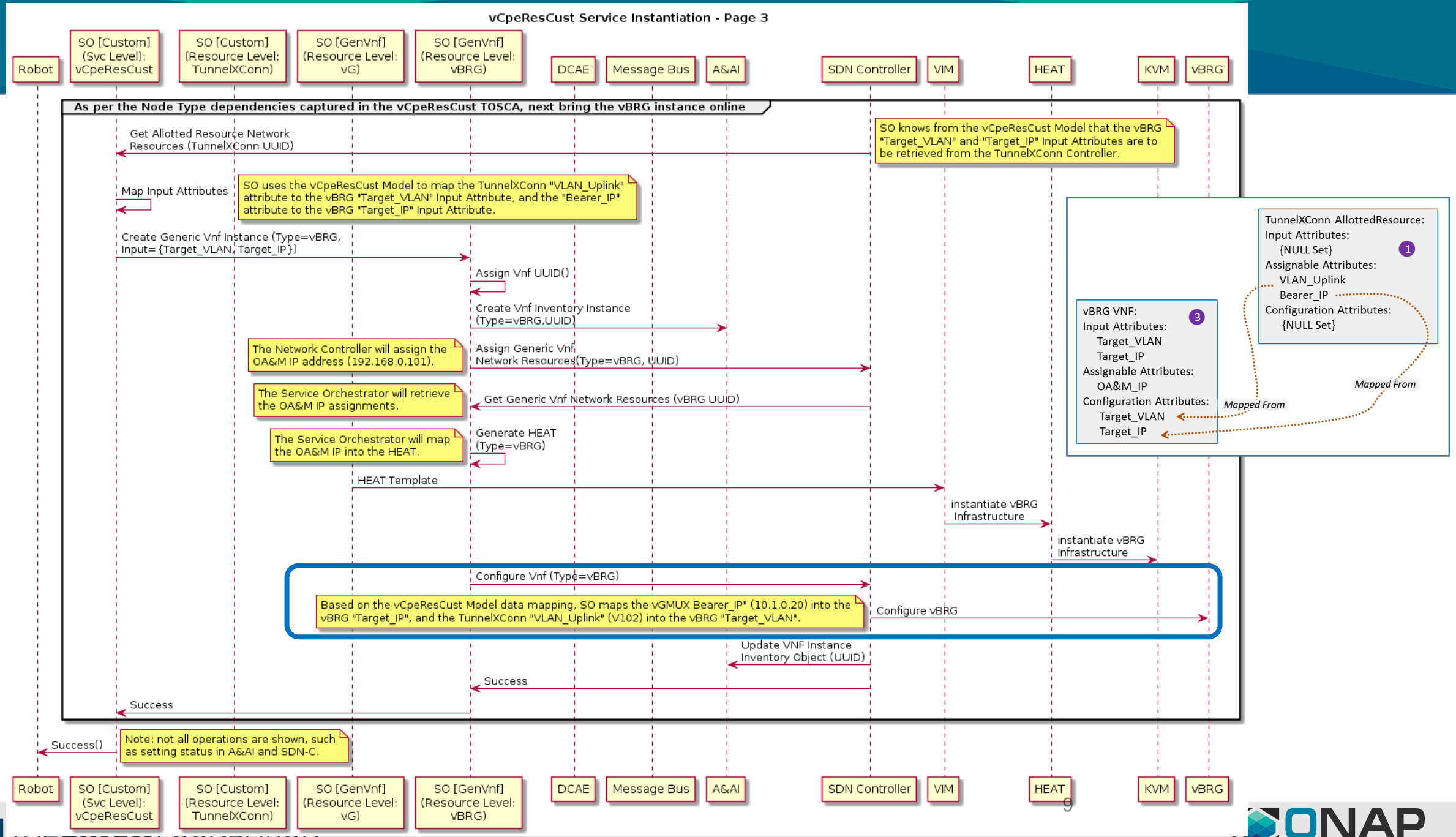


Customer service instantiation: vBRG

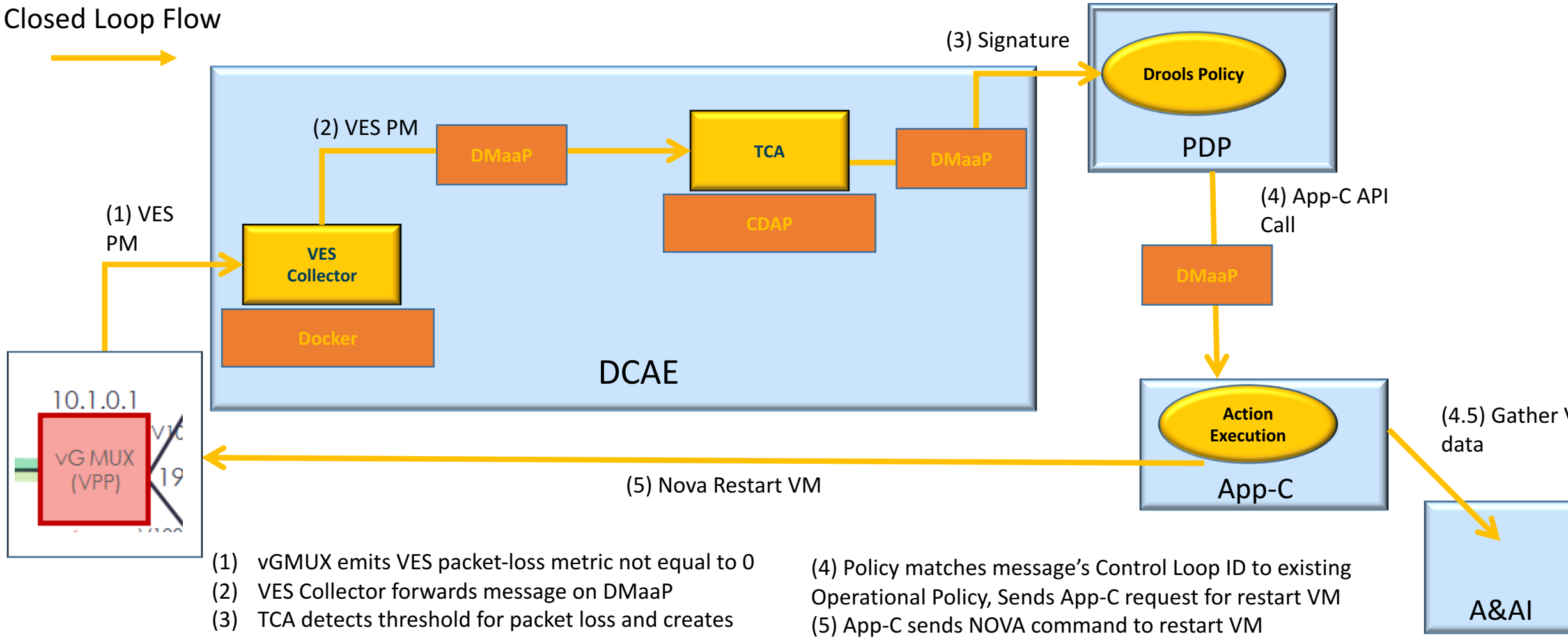
Set the following:

- vGMUX IP address
- VxLAN ID between BRG and vGMUX
- VLAN ID between vGMUX and vG




vCpeResCust Service Instantiation - Page 3
Robot
SO [Custom] (Svc Level): vCpeResCust
SO [Custom] (Resource Level: TunnelXConn)
SO [GenVnf] (Resource Level: vG)
SO [GenVnf] (Resource Level: vBRG)
DCAE
Message Bus
A&AI
SDN Controller
VIM
HEAT
KVM
vBRG
As per the Node Type dependencies captured in the vCpeResCust TOSCA, next bring the vBRG instance online
Get Allotted Resource Network Resources (TunnelXConn UUID)
SO knows from the vCpeResCust Model that the vBRG "Target_VLAN" and "Target_IP" Input Attributes are to be retrieved from the TunnelXConn Controller.
Map Input Attributes
SO uses the vCpeResCust Model to map the TunnelXConn "VLAN_Uplink" attribute to the vBRG "Target_VLAN" Input Attribute, and the "Bearer_IP" attribute to the vBRG "Target_IP" Input Attribute.
Create Generic Vnf Instance (Type=vBRG, Input={Target_VLAN, Target_IP})
Assign Vnf UUID()
Create Vnf Inventory Instance (Type=vBRG,UUID)
The Network Controller will assign the OA&M IP address (192.168.0.101).
Assign Generic Vnf Network Resources(Type=vBRG, UUID)
The Service Orchestrator will retrieve the OA&M IP assignments.
Get Generic Vnf Network Resources (vBRG UUID)
The Service Orchestrator will map the OA&M IP into the HEAT.
Generate HEAT (Type=vBRG)
HEAT Template
instantiate vBRG Infrastructure
instantiate vBRG Infrastructure
Configure Vnf (Type=vBRG)
Based on the vCpeResCust Model data mapping, SO maps the vGMUX Bearer_IP" (10.1.0.20) into the vBRG "Target_IP", and the TunnelXConn "VLAN_Uplink" (V102) into the vBRG "Target_VLAN".
Configure vBRG
Update VNF Instance Inventory Object (UUID)
Success
Success
Success()
Note: not all operations are shown, such as setting status in A&AI and SDN-C.
TunnelXConn AllottedResource:
Input Attributes:
{NULL Set}
1
Assignable Attributes:
VLAN_Uplink
Bearer_IP
Configuration Attributes:
{NULL Set}
vBRG VNF:
3
Input Attributes:
Target_VLAN
Target_IP
Assignable Attributes:
OA&M_IP
Configuration Attributes:
Target_VLAN
Target_IP
Mapped From
Mapped From

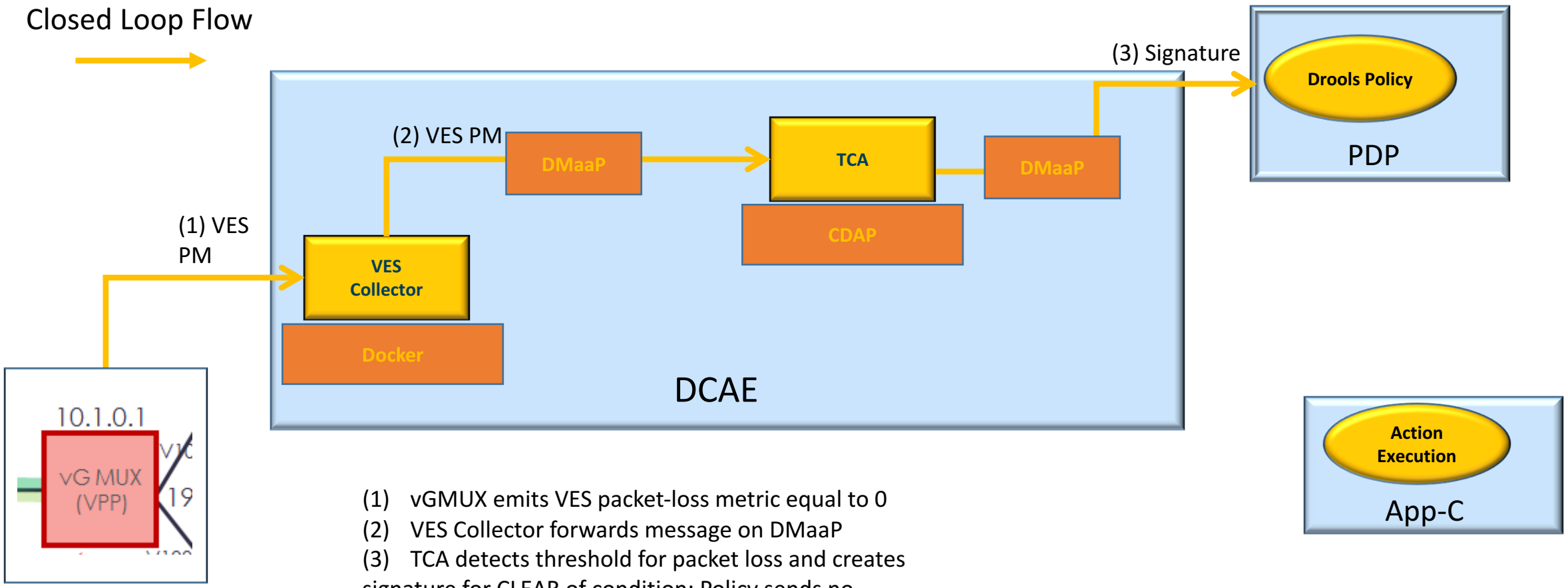
vCPE Runtime Flow R1 - ONSET



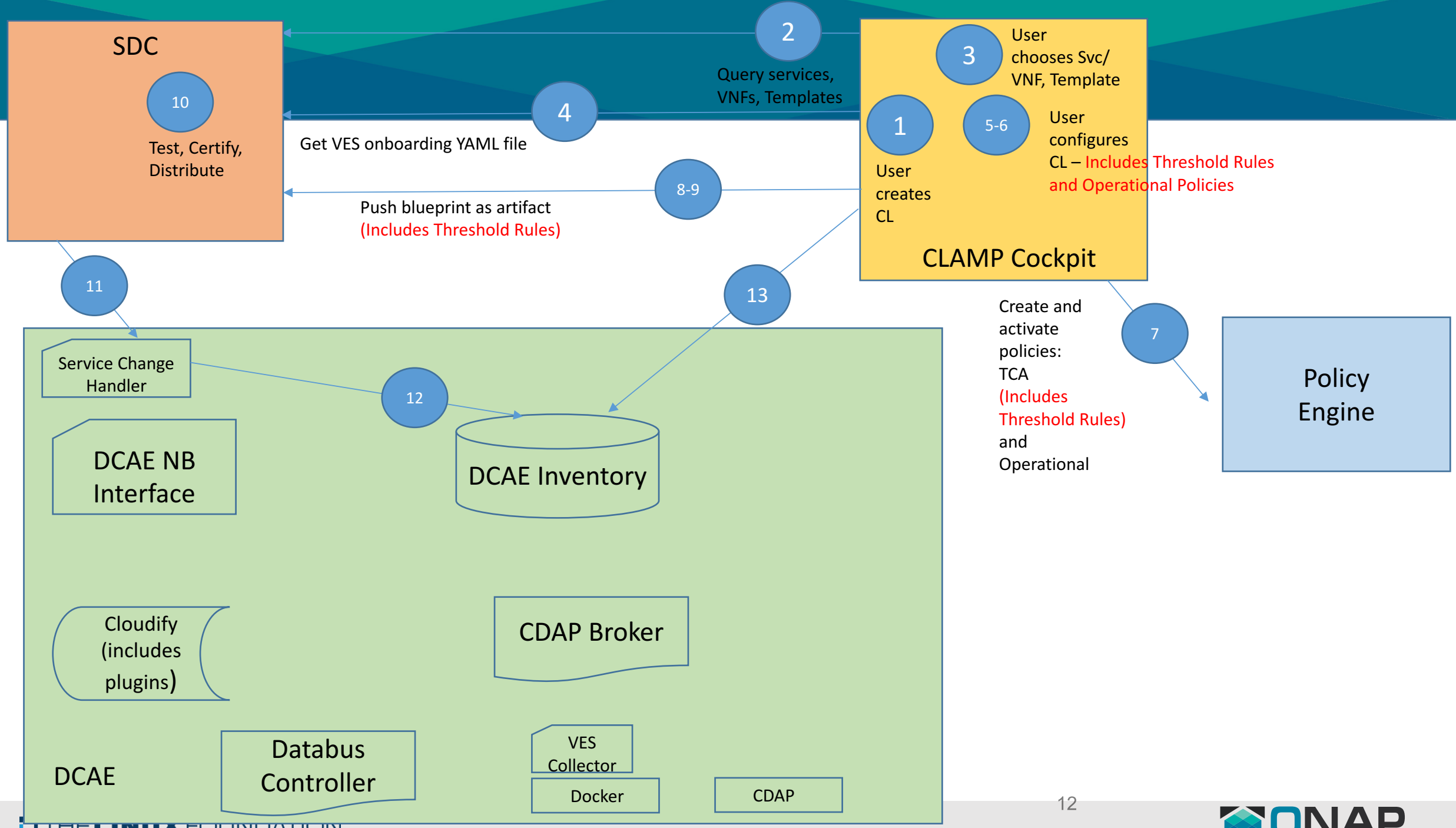
(1) vGMUX emits VES packet-loss metric not equal to 0
(2) VES Collector forwards message on DMaaP
(3) TCA detects threshold for packet loss and creates signature for ONSET of condition

(4) Policy matches message's Control Loop ID to existing Operational Policy, Sends App-C request for restart VM
(5) App-C sends NOVA command to restart VM

vCPE Runtime Flow R1 - CLEAR



(1) vGMUX emits VES packet-loss metric equal to 0
(2) VES Collector forwards message on DMaaP
(3) TCA detects threshold for packet loss and creates signature for CLEAR of condition; Policy sends no further actions to App-C


SDC
10
Test, Certify, Distribute
2
Query services, VNFs, Templates
4
Get VES onboarding YAML file
8-9
Push blueprint as artifact (Includes Threshold Rules)
3
User chooses Svc/ VNF, Template
1
User creates CL
5-6
User configures CL – Includes Threshold Rules and Operational Policies
CLAMP Cockpit
11
13
7
Create and activate policies: TCA (Includes Threshold Rules) and Operational
Policy Engine
Service Change Handler
12
DCAE NB Interface
DCAE Inventory
Cloudify (includes plugins)
CDAP Broker
DCAE
Databus Controller
VES Collector
Docker
CDAP

CLAMP Closed Loop Configuration

- Step 1: User Creates Closed Loop Model
- Step 2: CLAMP queries available service and resources in ASDC catalog. CLAMP also queries available closed loop TOSCA-based templates for configuration
- Step 3: User chooses vCPE Service and vGMUX Resource, and VES/TCA template
- Step 4: CLAMP downloads VES Onboarding YAML file for vGMUX VNF type

CLAMP Closed Loop Configuration

- Step 5: User goes to Operational Policy box, clicks, and creates a new Operational policy. Chooses chain of actions to execute for auto-healing of VM (Restart VM). Saves the policy as Signature1_OpsPolicy
- Step 6: User goes to TCA box, and defines the threshold rules, based on available KPIs in VES Onboarding YAML file (downloaded in Step 4). He associates each threshold configuration with the operational policy created earlier, Signature1_OpsPolicy

CLAMP Closed Loop Configuration

- Step 7: CLAMP creates separate policies
  - Operational (Drools) policy A: ClosedLoopControlName 11111
  - TCA Microservice policy referring to ClosedLoopControlName 11111
- Step 8: CLAMP generates blueprint based on template downloaded earlier from ASDC. This blueprint includes the correlation rules
- Step 9: CLAMP uploads blueprint to ASDC
- Step 10: ASDC Service is checked in, tested, certified and distributed
- Step 11: Distribution event is sent by ASDC over DMaaP and handled by Service Change Handler (SCH).

CLAMP Closed Loop Configuration

- Step 12: A new DCAE Service Type is created in Inventory; stored by Service UUID, resource UUID, artifact Name
- Step 13: CLAMP continually queries for distribution information about control loop
  - Query DCAE Service Types for an entry with:
    - Service Invariant UUID
    - Resource Invariant UUID
    - Artifact name
  - If an entry is found:
    - Transition Control Loop status to “Distributed”
    - Store Type ID (dcaeTypeId) for future use, associate it with Control Loop model

Thank You!