The VNF Challenges

- Lack of a standard format for VNF packaging

- This variation creates challenges for Service Providers that slow adoption:
  - Procurement
    - Complicated vendor engagement process
    - Lack of standard packaging or quality marks
  - Product Engineering
    - Acceptance testing/integration are time consuming and costly
    - Many tests must be repeated across multiple supplier VNFs
Is there a better model?
Vision

• Make it as easy to buy, sell, and onboard VNFs as it is smartphone apps by creating tools for VNF packaging and verification

• Support VNF Suppliers, Service Providers, and Integrators
Introducing ONAP VNF SDK

• Intended for use by Service Providers and VNF supplier DevOps teams

• Goal: Build an ecosystem for compatible VNFs
  - Package tools for VNF suppliers
  - Marketplace between suppliers and operators
  - Test framework to simplify acceptance testing (80/20 Rule)
    • Package validation
    • Functional tests
    • Lifecycle tests
VNF SDK Highlights

• Standard VNF packaging format
  - TOSCA-based, combining OPEN-O and ECOMP
• VNF SDK packaging tool for vendor CI/CD toolchain to simplify VNF packaging
• Vendor/Operator-neutral Marketplace framework ("app store")
• Validation and test tool

• Onboarding into ONAP
  - Could be extended for other orchestrators
The VNF Lifecycle…

**Retire**
- Deactivate
- Archive
- Decommission

**Manage**
- Monitor
- Update
- Upgrade

**Use (Services)**
- Request
- Instantiate
- Consume

**Deploy (Services)**
- Combine
- Assemble
- Configure

**Develop**
- Design
- Develop
- Test

**Deliver**
- Package
- Release

**Onboard**
- Validate
- Accept
- Catalog

---

* - Derived from TMF & ETSI/MANO lifecycle diagrams
VNF SDK in ONAP Beijing Release

- Resource Onboarding
- Service & Product Design
- Policy Creation & Validation

- Design Management Design
- Design Test & Certification

- Catalog

- CLAMP

- Design-Time (SDC)

- Run-Time

- Dashboard OA&M (VID)
- DCAE - Correlation Engine (Holmes)
- Service Orchestration Project
- A&AI/ESR

- Common Services
  - AAF
  - OOF
  - Logging
  - MUSIC
  - Others

- Multi-VIM/Cloud Infrastructure Adaptation Layer
- SDN-C (L0-L3 Controller)
- Application Controller (APPC) (L4-L7)
- Virtual Function Controller (VF-C) (Note 1)

- MSB/DMAAP

- External Systems
  - 3rd Party Controller
  - sVNFM
  - EMS

- Network Function Layer
  - Hypervisor / OS Layer
    - OpenStack
    - Commercial VIM
    - KBS
    - Public Cloud

- Private Edge Cloud
- MPLS
- Private DC Cloud
- IP
- Public Cloud

Recipe/Eng Rules & Policy Distribution

Note 1 - VF-C is ETSI-aligned.
VNFS SDK Packaging Model/Blueprint

- A declarative way to define deployment, operational and functional attributes of a VNF product
  - deployment time requirements and dependencies
  - telemetry indicator definitions
- TOSCA NFV simple profile
- TOSCA CSAR packaging format
Marketplace

- Provides a central place for uploading/downloading VNFs
- Validates VNF packaging
- Provides hooks to call other tools or libraries including validation and function tests
- Integrates with ONAP SDC for VNF Onboarding
Usage example: Bundling VNFs


- Put all the necessary files in a directory, in this example, we put them in the directory of name “csar_source”
  - Virtual machine images for VDUs
  - Scripts for installing and configuring each VDU per the standard TOSCA lifecycle workflows
  - Change history files, manifest file, license files, test files (Recommended but optional in the Amsterdam release). See ETSI GS NFV-SOL 004 for details.
<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute-profiles.yaml</td>
<td>Tosca template (blueprint)</td>
</tr>
<tr>
<td>pingpong.yaml</td>
<td>Change of history file</td>
</tr>
<tr>
<td>vnfsdk-profile-1.0.yaml</td>
<td>Directory containing VM images defined in tosca template</td>
</tr>
<tr>
<td>history.txt</td>
<td>Directory containing license information</td>
</tr>
<tr>
<td>images</td>
<td>Package manifest file</td>
</tr>
<tr>
<td>licenses</td>
<td>Scripts for installing and configuring each VDU</td>
</tr>
<tr>
<td>Tests</td>
<td>Directory containing test information</td>
</tr>
</tbody>
</table>
Create VNF Package

• Install VNFSDK package tools
  
  $ git clone http://gerrit.onap.org/r/vnfsdk/pkgtools
  $ cd pkgtools
  $ pip install -r requirements.txt
  $ pip install .

• Use VNFSDK package tool to create the VNF package
  
  $ vnfsdk csar-create -d DESTINATION [--manifest MANIFEST] [--history HISTORY] [--tests TESTS] [--licenses LICENSES] source entry
  
  -d DESTINATION: output CSAR file
  
  source: VNF package source directory
  
  entry: tosca entry definition file relative to directory “source”
Create VNF Package (cont.)

- VNFSDK package tool create the TOSCA-Metadata/TOSCA.meta file containing entry information for processing the output csar file.
Welcome to VNF Marketplace

Everything starts from here

Sort by Please select

- clearwater_huawei
  - Huawei Service DNS
  - Type: NSAR
  - Downloads: 18

- clearwater_huawei
  - Huawei Service
  - Type: NSAR
  - Downloads: 10

- clearwater_huawei
  - Huawei VNF Test Service
  - Type: NSAR
  - Downloads: 6
Upload Package
On-Boarding Progress

Welcome to VNF Marketplace

clearwater_ns

Onboarding Process

1. Validation Step 1 Execute
2. Validation Step 2 Execute
3. Validation Step 3 Execute
4. Validation Step 4 Execute

- LifeCycle Step 1 Execute
- LifeCycle Step 2 Execute
- LifeCycle Step 3 Execute

Check Package exists
- Download Package from Repository
- Execute Function Testing
### Onboarding Report

<table>
<thead>
<tr>
<th>Step</th>
<th>Parameter</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validation step-1</td>
<td>Validation Parameter1</td>
<td>pass</td>
</tr>
<tr>
<td>Validation step-2</td>
<td>Validation Parameter2</td>
<td>pass</td>
</tr>
<tr>
<td>Validation step-3</td>
<td>Validation Parameter3</td>
<td>pass</td>
</tr>
</tbody>
</table>
Welcome to VNF Marketplace

Everything starts from here

<table>
<thead>
<tr>
<th>Name</th>
<th>Provider</th>
<th>Description</th>
<th>Type</th>
<th>Size</th>
<th>Created at</th>
<th>Last Modified at</th>
<th>Downloads</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>clearwater_huawei</td>
<td>Huawei</td>
<td>Huawei Service DNS</td>
<td>NSAR</td>
<td>68.5 KB</td>
<td>2017-03-25 5:57 AM</td>
<td>2017-03-25 5:57 AM</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>clearwater_huawei</td>
<td>Huawei</td>
<td>Huawei Service</td>
<td>NSAR</td>
<td>68.5 KB</td>
<td>2017-03-25 5:57 AM</td>
<td>2017-03-25 5:57 AM</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>clearwater_huawei</td>
<td>Huawei</td>
<td>Huawei VNF Test Service</td>
<td>NSAR</td>
<td>68.5 KB</td>
<td>2017-03-26 5:38 AM</td>
<td>2017-03-26 5:38 AM</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>clearwater_huawei</td>
<td>Huawei</td>
<td>VENDOR-X Service</td>
<td>NSAR</td>
<td>68.5 KB</td>
<td>2017-03-26 5:45 AM</td>
<td>2017-03-26 5:45 AM</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>test_vnf</td>
<td>Huawei</td>
<td></td>
<td>NSAR</td>
<td>58.5 KB</td>
<td>2017-03-27 1:13 PM</td>
<td>2017-03-27 1:13 PM</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
VNF Onboarding

• ONAP Onboarding is handled by Service Design and Creation

• VNF SDK integrated with SDC
  - After processing, verified VNFs are available in the VNF SDK Marketplace

• SDC connects to the VNFSDK marketplace via RESTful API to view and onboard VNFs
Next steps

• Adding AT&T ICE scripts for HEAT-based VNF support

• Preparing for Compliance/Verification Program
  - Integrating with OPNFV Dovetail test framework
  - Supporting ONAP VNF Requirements
  - Working at LFN level to prepare for VNF compliance testing

• Functional tests/Lifecycle tests
Takeaways

• VNF Suppliers
  - Download our tools and start using them in your environment

• Service Providers and System Integrators
  - Use VNFSDK for pre-onboarding vendor engagement
  - Share requirements or tests that should be included in a verification program

• Entrepreneurs
  - Consider third-party marketplace