Managing Tailor-Made Enhanced Packet Cores for 4G/5G Testbeds in OSM with the SimulaMet OpenAirInterface VNF

Thomas Dreibholz, dreibh@simula.no
Andrés Felipe Ocampo, andres@simula.no

OSM10 Hackfest
December 2, 2020
Table of Contents

- OpenAirInterface and Our Goal
- Basic Testbed Setup
- The SimulaMet EPC VNF
- Juju Configuration and Challenges
- Managing Builds
- Demo
OpenAirInterface (OAI)

- OpenAirInterface (OAI):
  - Open Source software for EPC and eNodeB (i.e. packet core and base stations)
  - Details: https://www.openairinterface.org
  - 4G LTE available, 5G under development
  - Ongoing work, with many different Git branches

- Idea:
  - Manage OAI setups in OSM (at least, the EPC part)
    - Automatic setup and deployment
  - Easy to add additional features (e.g. Mobile Edge Computing components)
  - Open Source, of course! → https://github.com/simula/5gvinni-oai-ns
Setting Up a 4G/5G Testbed

- **Hardware:**
  - User Equipment (modems, smartphones, etc.)
  - Programmable sim cards
  - Software-Defined Radio boards

- **For the rest (eNodeBs, EPC):**
  - OpenAirInterface Open Source software
  - Running on regular Linux PCs
  - **But: difficult to install and maintain!**
Our Goal: An OpenAirInterface VNF

- Main purpose: testbed setups for research and development
- OAI EPC as VNF
  - Easy to use, EPC should (hopefully) work “out of the box”
  - Build of OAI software inside VMs, according to specified Git repositories and commits ⇒ get exactly the desired installation
- NSs using the VNF and possibly other VNFs
  - Example 1: add Mobile Edge Computing services to EPC
  - Example 2: get basic EPC to test extended eNodeB software
  - ...
Basic Testbed Setup

OpenSource MANO

Juju containers
- MME
- HSS
- SPGW-C
- SPGW-U

UEs eNodeBs

S1-C
S1-U
SGi

The Internet

PDN

MME
S6a
S11
HSS

sysstat

SPGW-U
EPC
SPGW-C

SimulaMet EPC NS

OpenStack Cluster
What is needed for the VNF?

- Base VDU image
- The VNF itself
- Juju Charms to configure the components
- Management of the build process
Base VDU Image

- VDU image goals:
  - Full-featured base VDU image, including development and debug tools
  - Different versions of Ubuntu LTS (Xenial, Bionic, Focal)
  - Up-to-date (i.e. all updates installed)

- Packer scripting:
  - Fully automatic installation using Packer
  - Preseeding (Ubuntu < 20.04); Subiquity (Ubuntu ≥ 20.04)
    - Additional PPA, keyboard layout setup, EFI boot, etc.
    - All updates installed
    - => Fresh, state-of-the art installation (avoids issues with “old” installations)
  - Details: [https://github.com/simula/nornet-vmimage-builder-scripts](https://github.com/simula/nornet-vmimage-builder-scripts)
The SimulaMet EPC VNF

HSS: Home Subscriber Server
MME: Mobile Management Entity
SPGW-C: Control Plane of the Packet Data Network Gateway
SPGW-U: User Plane of the Packet Data Network Gateway
VNF Parameters Example

- # ====== HSS ===============================================
  hss_git_repository: 'https://github.com/simula/openairinterface-openair-hss.git'
  hss_git_commit: 'dreibh/cassandra-build-fix-2020.w44'
  hss_S6a_address: '172.16.6.129'
  network_realm: 'simula.nornet'
  network_k: '449C4B91AEACD0ACE182CF3A5A72BFA1'
  network_op: '1006020F0A478BF6B699F15C062E42B3'
  network_imsi_first: '2428881234500000'
  network_msisdn_first: '24288880000000'
  network_users: '1024'

- # ====== MME ==============================================
  mme_git_repository: 'https://github.com/simula/openairinterface-openair-mme.git'
  mme_git_commit: '2020.w44'
  mme_S1C_ipv4_interface: '192.168.247.102/24'
  mme_S1C_ipv4_gateway: '0.0.0.0'
  mme_S11_ipv4_interface: '
  mme_S1C_ipv6_gateway: '
  mme_S11_ipv4_interface: '172.16.1.102/24'
  mme_S6a_address: '172.16.6.2'
  network_mcc: '242'
  network_mnc: '88'
  ...

Tailor-Made EPC Setup according to Your needs!
Configuration with Juju

- Day-0/1: For each VDU (EPC component, i.e. HSS, MME, SPGW-C, SPGW-U):
  - Install necessary additional packages (depends on component)
  - Set up network configuration
  - Clone component sources (Git repository and commit)
  - Build the sources
  - Create/update component's configuration files
  - Write systemd unit file (for “sudo service <component> start|stop|restart”)
  - Some convenience: System-Info, login banner per component, prompt colours, etc.
  - Start the component

- Day-2: actions to start/stop/reconfigure components
Juju Proxy Charm Challenges

- First version: charms.sshproxy._run() with lots of escaping
  - $ ⇔ \$ ; \ ⇔ \ \ ; " ⇔ \" ; " ⇔ \\
  - Really ugly to read, a mess to debug, ...

- Better solution:
  - Write “normal” code => Base64 encoding => charms.sshproxy._run()
  - => echo “<Base64 string>” | base64 -d | /bin/bash -x

- Even better:
  - Some helper functions as a Python library “VDUHelper”
    - Code execution in VDU, file upload into VDU, etc.
  - Add same library code to all our Juju proxy charms (=> automatically, with CMake ...)
Managing VNFD/NSD Builds

- Multiple manual steps to generate and deploy VNFs and NSs
  - Strictly verify all YAML files with yamllint (very useful, to avoid problems and surprises!)
  - Copy Charm files to VNFDs and build Charms (charm build ...)
    - Including our library “VDUHelper”
  - Verify descriptor(s) and generate VNFD package(s)
    (validate_descriptor.py, generate_descriptor_pkg.sh)
  - Verify descriptor(s) and generate NSD package(s)
- Initial approach: write a Makefile
- Better approach:
  - Git for source management ⇨ information about all relevant source files
  - Let CMake write Makefiles and take care of dependencies!
Live Demo: SimulaMet EPC + Cloud RAN
Scenario Overview
Cloud RAN
Network Setup
Murphy’s Law

Murphy’s Law: „Anything that can go wrong, will go wrong.“

„Accident ferroviaire de la gare Montparnasse“
Image sources: Wikimedia
OSM Wishlist

- OSM 8.0.2 is a great stability improvement!
  - Only issue so far, with multiple NS instances: steadily increasing number of osm-mon-collector processes, until OOM killer starts terminating them → bug in OSM?

- Wishlist items:
  - Improvement for OSM (and Juju) dependencies:
    - OSM and Juju depend on many APT packages, PIP Python packages, Snap, …
    - Murphy’s Law → many opportunities for things to go wrong
    - In case of any dependency issues, OSM installation/NS instantiation is going to fail (happened a couple of times in the past, particularly due to PIP package issues)
      - => Wishlist: possibility to only depend on APT packages?
  - If possible: feature to freeze versions
    - => Wishlist: possibility to even run without any Internet access?
Sources

- Get the sources here: https://github.com/simula/5gvinni-oai-ns
  - Open Source, GPL-licensed
  - README: how to set up a testbed
  - `vmimage-builder-scripts/` (submodule): VDU preseeded image build script
  - `juju/`: The Juju Charms used by the VNF
  - `SimulaMet-OAI-EPC_vnfd/`: VNF descriptor
  - `SimulaMet-OAI-EPC_nsd/`: NS descriptor for simple example
Any Questions?

Thomas Dreibholz

dreibh@simula.no

https://www.simula.no/people/dreibh

Andrés Felipe Ocampo

andres@simula.no

https://www.simula.no/people/andres

This project has received funding from the EU’s Horizon 2020 research and innovation programme under grant agreement No 815279.