Scalable Architecture for Service Function Chain Orchestration

Sahel Sahhaf, Wouter Tavernier, Janos Czentye, Balazs Sonkoly Pontus Skoldstrom, David Jocha, Jokin Garay
30/09/2015- EWSDN 2015
Outline

- Introduction to SFC/NFV
- SFC and service decomposition
- SFC orchestration
  - Requirements of orchestration
  - ESCAPE framework
  - Embedding algorithm
- Performance evaluation
- Enhancements towards a scalable orchestrator
Service Function Chaining

- **Service Function Chaining** enables composing services out of multiple Network Functions (NFs) represented by a Service Graph (SG)
  - Example NFs: Firewalls, NAT, TCP optimizers, web proxies
  - Traditionally deployed as **hardware appliances**

- **Network Function Virtualization** (NFV) enables NFs to run in **software** on low cost off-the-shelf server hardware.
  - Simplifies service chaining and provisioning process
  - Reduces the cost
  - Introduces new services
Service decomposition in SFC

Concept:

- Step-wise translation of high-level NFs into more refined NFs
- Decomposition of a compound NF into more elementary/atomic NFs
  - Elementary NFs can be mapped to the infrastructure

Example:

- Intrusion Detection Service
Advantages of service decomposition

- Reuse of existing components to build new modules
  - Increased development speed and reduced development costs
- Multiple implementation of the same functionality
  - A generic NF type can be mapped to different platforms (e.g. x86, ARM, OF switch, etc)
  - Implementation can be optimized for different aspects (using more CPU but less storage, using less CPU but more storage, etc)
- Requesting generic services
  - No concern about detailed implementation and available resources
- Better resource usage
  - Load balancing to the appropriate infrastructure components
- Advanced resource optimization by jointly optimizing decomposition and resource allocation (embedding)
Service chain orchestration

- An orchestrator is responsible for the service management and orchestration

- **Main functionalities:**
  - Optimal mapping of VNFs across infrastructure
  - Instantiating VNFs at reasonable locations
  - Keeping track of VNFs location
  - Assigning and scaling resources to the VNFs
  - Service VNFs monitoring
Service chain orchestration - requirements

Orchestration process is impacted by scale of Telecom operator network and the number of service requests.

- **Telecom operator network**
  - Hierarchical structure
  - Dense core router meshed network (consisting of inner and outer core Points of Presence (PoPs))
  - End customers are interconnected to this core network via a hierarchy of tree-structured access- and metro aggregation networks.
Service chain orchestration - requirements

- Telecom operator network (e.g. BT)

sources: http://www.kitz.co.uk/adsl/21cn_network.htm

- 50K devices at different parts of the network
- 10M devices including CPEs to be orchestrated
- 2.5 new customers per day
- 5-10K service requests per day
Service chain orchestration – prototype framework

- ESCAPE framework: supports the development of several parts of the service chaining architecture
  - VNF implementation
  - Traffic steering
  - Virtual network embedding
Service chain orchestration – prototype framework

- ESCAPE
  - Service layer
  - Orchestration layer
  - Infrastructure layer

Implemented in Python on top of POX (OpenFlow controller) platform and Mininet
Service chain orchestration – prototype framework

- **Service layer:**
  - contains an API and a GUI
  - users can request and manage services and NFs through the GUI
  - API get the SG request and passes that to a dedicated service orchestrator
  - A mapping of service to the infrastructure nodes is performed by the SG mapper module
Service chain orchestration – prototype framework

- **Orchestration layer:**
  - Requests arrive from the service layer are forwarded to the RO
  - RO is the main entity which maps the requests to the abstract domain view provided by the Domain virtualizer
  - RO collects and forwards all required data to RO mapper
    - The request
    - The domain view
    - NF-IB

- **Controller adapter**
  - Builds an abstract domain view based on the gathered technology specific information of the resources
  - Delegates requests to the corresponding adapter
Service chain orchestration – prototype framework

- Network Function Information Base (NF-IB)
  - stores the NF models/abstractions, NF relationships, NF implementation image(s) and NF resource requirements
  - supports the definition of abstract NFs such as a FireWall, referring to a type, a potential number of ports/interfaces, as well as dependencies to other NFs
  - stores the NF relationships into a tree-like data structure in support of the decomposition process

- NF-IB is implemented in Neo4j database
  - stores key-value pairs for nodes and edges
  - for each NF, the tree-like structure with all the corresponding information of nodes and links are stored
Service chain orchestration – embedding algorithm

**Embedding problem:** Map service requests composed of NFs to physical network

**Objective:**
- Minimize the cost of the mapping provisioning
- Increase the request acceptance ratio
- Fulfill the QoS requirement of the requests

**Assumptions:**
- Several decompositions for a service request
- Network Functions with different types
  - Virtualization techniques: Xen, Vmware, VirtualBox
  - Docker
  - Process in a container (Click process)
  - Packet I/O drivers: DPDK
  - Hardware appliances

**Proposed algorithm:**
- Decomposition selection: for each decomposition calculate a cost based on
  - Number of potential physical nodes which can host the VNFs of the decomposition
  - Number of VNFs in the decomposition
  - Cluster factor of the decomposition (how same-type VNFs are connected)

- Mapping: backtracking mechanism


S. Sahhaf, W. Tavernier, M. Rost, S. Schmid, D. Colle, M. Pickavet, P. Demeester, “Network service chaining with optimized network function embedding supporting service decompositions”, Computer Networks
Performance evaluation

- **Simulation setup**
  - Physical networks
    - Random regular networks with degree 3, 100-1000 nodes
    - Node capacity = [100-300]
    - Link bandwidth = [100-300]
    - Link delay = [10-50]
  - Service requests (SGs)
    - Random DAGs
    - NF demands = [1-20]
    - Link demands = [1-20]
    - Nr of NFs in SGs = 5,10
    - Nr of decompositions per NF = 2,3,4
Performance evaluation

Impact of number of NFs/SG in different network size

Impact of number of decomp/NF in different network size
Towards a scalable orchestrator

‘read dcmp’ block is the most time consuming block in smaller topologies while ‘select dcmp’ block is a major issue in larger topologies

- Parallel/distributed embedding
  - Costly calculations are done in parallel
  - Neo4j HA can be used to distribute the full database onto multiple nodes to parallelize the ‘read dcmp’ block
  - ‘select dcmp’ can be calculated in parallel as the cost calculation of each decomposition is independent of others
  - Parallelizing the ‘map’ block is challenging:
    - Considering all possible combination of NFs mapping to physical network nodes => small scale (o(100))
    - Mapping NFs and calculating paths in parallel => challenge is to avoid threads reserving the same resource. (Batch scheduling- get dedicated access to the resources.)
Towards a scalable orchestrator

- Hierarchical embedding
  - Service graph is divided to subgraphs using the decompositions available in the NF-IB
  - Each subgraph can be given to a different infrastructure domain to be orchestrated locally
  - Challenge is the amount of the resource information to be exposed to the upper layer orchestrator to enable efficient orchestration

- Pre-defined service chains
  - have pre-defined service chains with pre-defined decomposition templates
  - different parts of the embedding can be processed in advance, before the service is requested
Summary

- In Service Function Chaining (SFC), virtualized Network Functions (NFs) are chained to compose a network service.
- A resource orchestrator to steer the control of SFCs is required to map the NFs of a requested service to the infrastructure network and compute resources.
- Orchestration might involve thousands of requests in one day to be mapped on one or more infrastructure provider networks involving ten thousands of network elements and thus **scalability** is an important characteristic of an orchestrator component.
- As a proof of concept a mapping algorithm supporting service decomposition was implemented and the key time consumers within the implemented PoC were identified.
- **Potential enhancements towards a scalable orchestrator:**
  - Parallel/ distributed embedding
  - Hierarchical embedding
  - Pre-defined service chains
Thank you!