On the Necessity of Accounting for Resiliency in SFC

Ghada Moualla, Thierry Turletti, Mathieu Bouet, Damien Saucez

On the Necessity of Accounting for Resiliency in SFC

Ghada Moualla*, Thierry Turletti*, Mathieu Bouet†, Damien Saucez*
∗ Inria Sophia Antipolis – France
† Thales Communications & Security – France

***Abstract*—When deploying network service function chains the focus is usually given on metrics such as the cost, the latency, or the energy and it is assumed that the underlying cloud infrastructure provides resiliency mechanisms to handle with the disruptions occurring in the physical infrastructure. In this position paper, we advocate that while usual performance metrics are essential to decide on the deployment of network service function chains, the notion of resiliency should not be neglected as the choice of virtual-to-physical placement may dramatically improve the ability of the service chains to handle with failures of the infrastructure without requiring complex resiliency mechanisms.**

I. Introduction

To provide the services demanded by their customers, network providers must continually purchase, deploy, and reconfigure middle-boxes, which results in high CAPEX and OPEX and leads to long product cycles to provide these services with a strong dependence on specialized hardware. To provision more rapidly new services while reducing costs, *Network Functions Virtualization* (NFV) was proposed [1] and extended by *Service Function Chaining* (SFC) that combines multiple network functions in specific orders. NFV and SFC leverage virtualization to deploy services on commodity hardware hence reducing costs but raising the new challenge of how to provide efficiency and resiliency into software with shared and error-prone hardware resources [2].

To realize service chains, the placement of the virtual functions onto the physical infrastructure becomes critical as it plays a major role in the performance and robustness of the chain. However, while tremendous efforts have been realized on placing functions to reduce costs and improve performances [3], robustness is often neglected under the cover that the orchestrator can cope with failures. With this paper, we advocate that accounting for robustness when placing functions can significantly improve robustness of the chain without increasing the load of the orchestrator. To that aim, we study a reference chain and demonstrate with an exhaustive study how placement in the physical infrastructure can influence the overall robustness of the system even when the virtual chain itself is supposed to be robust to failures.

II. related work

Knowing that with virtualization the failure of a host propagates to all its Virtual Machines (VMs), robust placements of the VMs for critical services need to be considered carefully. Aware of this problem, Machida et al. propose an algorithm for the placement of redundant VMs based on affinity and anti-affinity rules to minimize the number of hosts needed to ensure a given level of redundancy [4].

Alternatively, Jayasinghe et al. present a placement algorithm that takes an application description with constraints and a datacenter description and translates them into tree models [5]. Their solution places VMs on physical machines by grouping VMs and placing groups on server clusters, then checking if the individual VM requirements are met. Also, this work does not discuss the robustness issue.

Furthermore, Mehraghdam et al. formalize service chaining requests using context-free language and present a mixed integer quadratic programming placement strategy [6]. In their evaluation, they maximize link available bandwidth and minimize latency and number of hosts but do not account for robustness.

Schöller et al. describe a deployment function that takes an abstract service description including placement and resiliency requirements as input and focusing on delay between redundant instances [7]. This deployment function is based on OpenStack using availability zones. Finally, Oechsner and Ripke discuss the topic of VM placement in the context of NFV deployment [8]. They utilize a placement mechanism with a resilience pattern mapped to OpenStack in order to provide an automatic deployment of resilient components in cloud environments. The considered use-case is to place a redundant active-backup pair of VMs with the requirement of placing instances close enough to ensure the end-to-end delay, but far enough apart to guarantee a certain level of availability. Their heuristic takes as an input the availability of the components of the physical infrastructure, the delay between them, the maximum delay between the redundant instances, and the minimum availability of the joint component.

In the light of the current research, we present a general discussion on the necessity of considering resiliency while placing virtual functions and show that the choice of the placement may have a dramatic impact on the ability of the system to be robust to failures.

III. Service function chain robustness

A. Reference environment

To illustrate the impact of the network functions placement on the resiliency of a chain, we consider the reference chain presented in Fig. 1 and composed of two equal sub-chains of 3 functions such that when a flow must be processed by the chain, it can be processed by

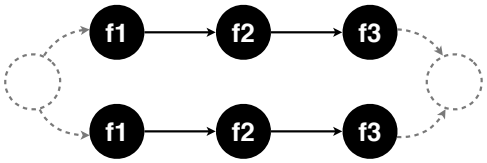
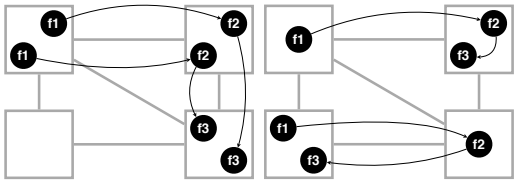


Figure 1. Reference chain.



(a) Not robust (b) Robust

Figure 2. Examples of mapping and their robustness to one failure.

either sub-chain. The sub-chain that processes a flow is selected by the cloud infrastructure (e.g., load balancer). This reference chain is general as it is at the same time robust and fragile. Robust as processing can be done by any of sub-chain and fragile as if an element in one sub-chain fails, the whole sub-chain is disrupted. This situation is common when flow state is mandatory. We assume that the system that decides the sub-chain to be used is able to determine if a sub-chain is working properly or not while selecting the sub-chain of a flow and that it functions properly.

Fig. 2 shows an example where the placement of the chain in the physical infrastructure can impact the robustness of the chain to one single failure (node or link). To study the impact of the placement of functions onto the physical infrastructure, we consider all placements of the reference chain onto the redundant tree[1] presented in Fig. 3. This topology is intentionally simple, contrary to more connected datacenter topologies such as fat tree, BCube etc., to clearly outline the impact of placement on the service robustness. As the number of potential placements is combinatorial, we randomly sampled the set of all solutions to take a total of 16,718 different placements[2].

To assess the topological properties of the different placements, we consider three usual metrics: (i) the *number of physical hosts* and (ii) the *number of Top-of-the-Rack (ToR) switches* of the physical infrastructure involved in the placement, that indicate how the placement shares the load and how it is distributed in the infrastructure, and (iii) the *number of virtual functions per host* that indicates how processing virtualization is leveraged by the placement.

Considering the topology as a graph, we can identify two categories of failures: the failure of a node (i.e., a computing host or a switch) or the failure of an edge (i.e., a link). To assess the robustness of the placement against failures, we independently considered these two categories and exhaustively tested every combination of n-failures of elements of the category (i.e., n nodes or n links). A placement is considered as n-robust if a flow can be processed by the service chain for any failure of n physical elements of the network. However, by construction of our chain, there exists always at least one case where the chain is not robust: if several failures happen simultaneously (i.e., $n \geq 2$). For this reason, we rather consider the probability of the chain to fail in case of n simultaneous failures when failures are independent and equiprobable.

[1]As the purpose is to motivate the robustness problem in service function chaining, we do not apply any constraint (e.g., bandwidth) on the placement.

[2]All the code and placements used in this paper are available at https://team.inria.fr/diana/files/2016/05/procon16.zip

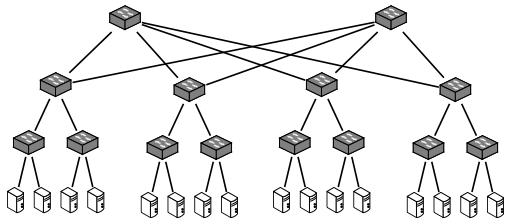


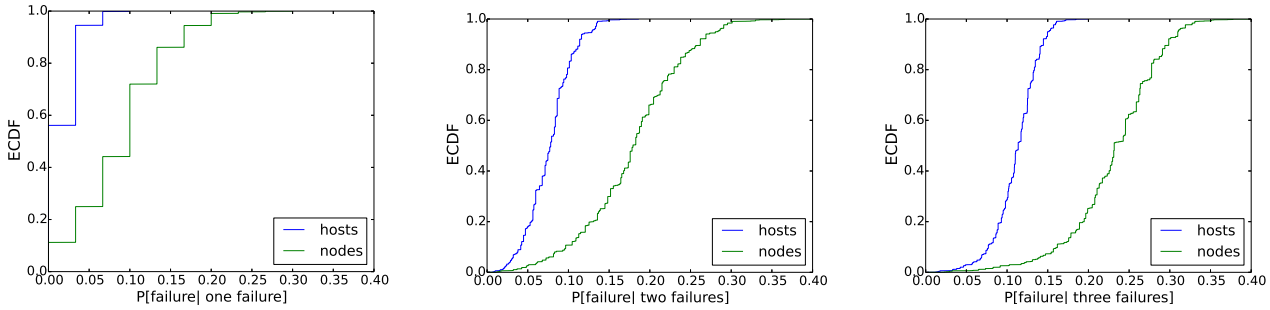
Figure 3. Reference topology.

B. Robustness analysis

Overall, no strong linear correlation exists between topological and placement properties in terms of robustness. Nevertheless, the probability that the chain fails is weakly correlated with the number of functions deployed on one physical host (correlation is 0.68) as it is sufficient that at least one function of each sub-chain is deployed on the same host to break the chain in case of one single failure. This conclusion is confirmed by Fig. 4 that shows the empirical cumulative distribution function of having a chain disruption in case of node failures. We consider the failure of a node in the topology in general (i.e., including switches and computing hosts) and the cases accounting only for the failure of a computing host. Fig. 4 indicates that the robustness of the chain is less sensitive to the failure of computing hosts than to the failure of switches. This is because each host is connected to the network without redundancy via a ToR switch. Therefore, the failure of one single ToR switch breaks the entire chain even if it is deployed on different physical hosts. On the contrary, to disrupt the chain with the failure of a host, it is necessary that at least one function of each sub-chain is deployed on the same physical host.

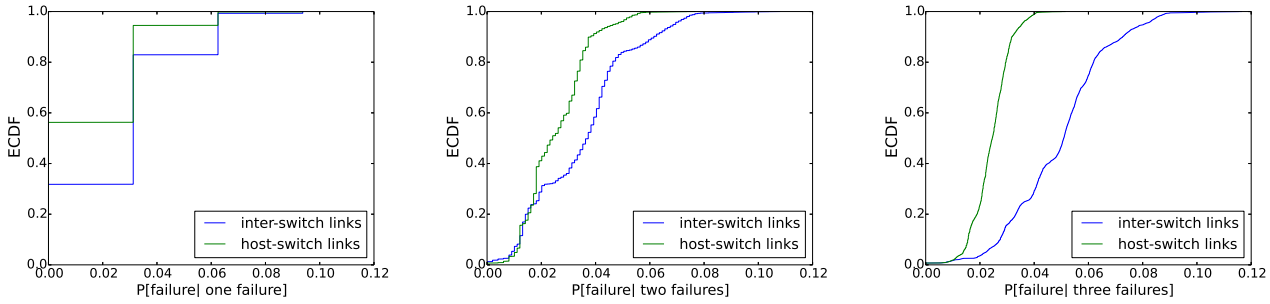
Similar conclusions can be drawn while considering link failures instead of node failures (see Fig. 5) where we distinguish failures of inter-switch links and host-switch links. We can observe that despite the redundancy of the topology, backbone link failures impact the robustness of the chain, particularly the failure of inter-switch links as actually losing the connectivity of ToR switches may impair the chain, regardless of the computing redundancy.

As illustrated by Fig. 4 and Fig. 5, the overall robustness behaviour is the same regardless of the number of simultaneous failures except that the likelihood of having a chain disruption increases with the number of failures.



(a) 1 failure (b) 2 simultaneous failures (c) 3 simultaneous failures

Figure 4. Probability of chain disruption in case of node failure.



(a) 1 failure (b) 2 simultaneous failures (c) 3 simultaneous failures

Figure 5. Probability of chain disruption in case of link failure.

IV. Discussion

In this paper, we advocate that the choice of the placement of virtual functions constituting chains onto a physical infrastructure should not neglect the robustness of the chain as while some placement may offer good performance, they may be very fragile to failures of a physical component. In Sec. III-B we exhaustively studied the impact on the robustness of the placements of a reference chain in a tree topology. We have seen that even though a chain is logically robust, if robustness is not accounted while deciding the placement it may be broken by a single failure.

One may argue that the failure of physical infrastructure elements is not an issue as recovery mechanisms are implemented by the orchestrator. What we answer is that if these mechanisms are mandatory to make the system truly resistant, they remain slow as they require VM migrations. On the contrary, a wise selection of placement permits to be robust to usual simple failures of the physical infrastructure without having to trigger migrations and thus reducing the stress on the orchestrator that would then be used only for complex situations. We plan on building new mechanisms for deploying service function chains with the use of not only the usual performance metrics, but also the resiliency metrics in order to minimize the probability of the service being disrupted.

Acknowledgements

This work is funded by the French ANR under the REFLEXION project (ANR-14-CE28-0019).

References

[1] ETSI, NFVISG, "Network Functions Virtualisation (NFV) Architectural Framework," *ETSI GS NFV*, vol. 2, no. 2, 2013.

[2] ETSI, ISGNFV, "ETSI GS NFV-REL 001 V1. 1.1: Network Functions Virtualisation(NFV); Resiliency Requirements," 2015.

[3] Y. Li and M. Chen, "Software-Defined Network Function Virtualization: A Survey," *IEEE Access*, vol. 3, pp. 2542–2553, 2015.

[4] F. Machida, M. Kawato, and Y. Maeno, "Redundant virtual machine placement for fault-tolerant consolidated server clusters," in *2010 IEEE Network Operations and Management Symposium - NOMS 2010*, April 2010, pp. 32–39.

[5] D. Jayasinghe, C. Pu, T. Eilam, M. Steinder, and I. Whalley, "Improving performance and availability of services hosted on iaas clouds with structural constraint-aware virtual machine placement," in *IEEE SCC*, 2011.

[6] S. Mehraghdam, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on*. IEEE, 2014, pp. 7–13.

[7] M. Schöller, M. Stiemerling, A. Ripke, and R. Bless, "Resilient deployment of virtual network functions," in *2013 5th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, Sept 2013, pp. 208–214.

[8] S. Oechsner and A. Ripke, "Flexible support of vnf placement functions in openstack," in *Network Softwarization (NetSoft), 2015 1st IEEE Conference on*, April 2015, pp. 1–6.