Global State, Local Decisions: Decentralized NFV for ISPs via enhanced SDN

Alberto Rodriguez-Natal, Vina Ermagan, Ariel Noy, Ajay Sahai, Gideon Kaempfer,
Sharon Barkai, Fabio Maino, Albert Cabellos-Aparicio

***Abstract*—The Network Function Virtualization (NFV) paradigm is rapidly gaining interest among Internet Service Providers (ISPs). However, the transition to this paradigm on ISP networks comes with a unique set of challenges, namely legacy equipment already in-place, heterogeneous traffic from multiple clients, and very large scalability requirements. In this paper we thoroughly analyze such challenges and discuss NFV design guidelines that address them efficiently. Particularly, we show that a decentralization of the NFV control while maintaining global state improves scalability, offers better per-flow decisions and simplifies the implementation of Virtual Network Functions (VNFs). Building on top of such principles, we propose a partially decentralized NFV architecture enabled via an enhanced Software Defined Networking (SDN) infrastructure. We also perform a qualitative analysis of the architecture to identify advantages and challenges. Finally, we determine the bottleneck component, based on the qualitative analysis, which we implement and benchmark in order to assess the feasibility of the architecture.**

I. Introduction

THE Network Function Virtualization (NFV) paradigm enables software-hardware decoupling, flexible deployment of network functions and dynamic service provisioning [1]. Traditionally, network functions (e.g. firewalls, DDoS filters, TCP optimizers, etc.) are deployed by means of purpose-specific hardware appliances. However, the NFV paradigm proposes to virtualize these functions via software in order to dynamically instantiate, move and destroy them. Complementary to NFV, the Software-Defined Networking (SDN) [2] paradigm has also gained traction in the industry. SDN advocates for decoupling the control-plane from the data-plane. A central SDN controller centralizes the control and remotely programs the data-plane devices. Interestingly, both NFV and SDN serve for network virtualization. While data-center and campus networks can use SDN to virtualize network forwarding, Internet Service Providers (ISPs) can use NFV to virtualize network functions.

ISPs deploy in-network functions to efficiently manage the traffic and offer value-added services to their costumers. In this context, NFV would help to reduce both capital and operational expenses by enabling easier and cheaper deployment and simplified management of network functions. However, bringing NFV to ISP networks present a unique set of challenges. In addition to performance, manageability, reliability, stability, and security [1], [3], an NFV solution for ISP networks has to consider their large size, the legacy networking hardware already in-place and the heterogeneous traffic generated by ISP's customers. Therefore, an NFV architecture for ISPs must offer a platform able to scale to a wide range of different workloads and network conditions, while remaining agnostic to the Virtual Network Function (VNF) types required to process the different kinds of traffic.

These requirements dramatically increase the complexity of centralized control. Therefore, we suggest that typical NFV approaches with centralized control fall short for the ISP scale. Those solutions tend to leverage on SDN approaches that centralize -logically- both the state and the control. We advocate that, while the network state should be centralized, the control decisions must be decentralized and made locally.

In this paper, we analyze these requirements and propose a set of design guidelines to address them. Based on these principles, we describe a novel decentralized architecture that offloads part of the control to an enhanced SDN infrastructure and that federates local state through a global database. This makes possible to make efficient per-flow local decisions but based on global knowledge. Therefore, VNFs and client flows can be elastically accommodated. The enhanced SDN is enabled by collocating NFV modules within the SDN controllers and then pushing the controllers close to the data-plane devices they control.

Along with the architecture we present a qualitative analysis that highlights its advantages and challenges. To assess the feasibility of the architecture, we identify its potential bottleneck and provide a possible implementation that we support with experimental performance results.

II. Scenario requirements

In addition to common NFV requirements [1], [3], an NFV solution for ISP networks needs to consider the following:

a) Legacy hardware: Usually, ISP networks are long-run deployments where there has been a significant investment in network equipment. Contrary to enterprise or public IaaS cloud datacenters which in many cases are greenfield deployments, the networking hardware of ISPs is already in place and in most cases it is not simple to update, upgrade or replace. The architecture should remain agnostic to the underlaying devices and thus be able to be deployed on top of current networks.

A. Rodriguez-Natal and A. Cabellos-Aparicio are with the Technical University of Catalonia, Barcelona, Spain. E-mail: {arnatal, acabello}@ac.upc.edu

V. Ermagan and F. Maino are with Cisco Systems, San Jose, CA, USA. E-mail: {vermagan, fmaino}@cisco.com

A. Noy, A. Sahai, G. Kaempfer and S. Barkai are with Hewlett Packard Enterprise, Sunnyvale, CA, USA.. E-mail: {ariel.noy,ajay.sahai,gidi,sharon.barkai}@hpe.com

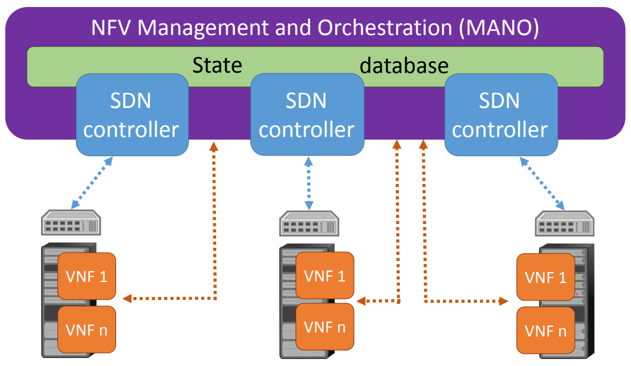

Fig. 1. **Common centralized NFV approach**

b) Traffic heterogeneity: As opposed to the traffic observed in datacenter networks, the expected traffic in an ISP network will likely come from a wide range of costumers and presents diverse characteristics. Such traffic will require different kinds of processing by large sets of heterogeneous network functions.

c) Number of flows: Due to the size of ISP deployments, achieving scalable fine-grain flow processing becomes a major challenge. These networks are typically very large and comprise millions of users. To individually manage large numbers of flows, the architecture needs to scale-out smoothly.

III. Global state, local decisions

With the advent of SDN and the possibilities of control-data decoupling, network architectures usually centralize the control to ease network deployments. However, we believe that the major challenge of an NFV deployment for ISP networks is that their scale, in terms both of traffic and number of independent subscribers, increases the complexity of keeping a scalable -logically- centralized control.

Existing works on SDN [4], [5] propose a distributed control. We take that approach further advocating for an NFV architecture where the control is not only distributed but also partially decentralized. We seek to find an optimal middle ground, between the decentralization of legacy networks and the centralization brought by SDN.

We argue that this optimal balance can be achieved by enforcing local control while keeping global state. That is, federating the state generated locally to make the outcome of the local decisions globally available. As a result, the decision on how to act on a flow can be made by the node processing the flow, but taking into account the global state of the system at that time.

For instance, assume a scenario where local controllers locally monitor the load of VNFs. To alleviate the load on a certain VNF, a controller may locally decide to steer some flows to a less loaded VNF. Via a global state database it knows the load of other instances of that VNF and can locally choose a less loaded one. Then, it publishes this decisions on the global state database. Thanks to this, if another controller has to locally process those flows, it knows that has to forward them to the new VNF.

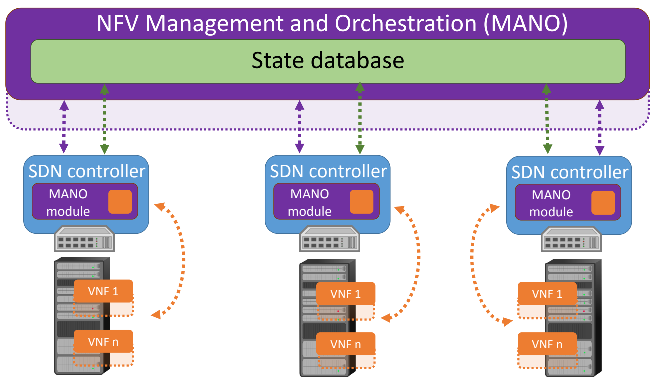

Fig. 2. **Decentralized NFV based on an enhanced SDN**

IV. Design principles

In what follows we propose a set of design guidelines to achieve this partial decentralization as well as to face the other requirements (mentioned in Section II) of the ISP scenario.

a) MANO disassembling: We refer to the ETSI architectural framework [6] that defines the Management and Orchestration (MANO) system as the central point for NFV control [7]. The MANO system comprises the global orchestration of the architecture and the management of the VNFs and the virtualized infrastructure. Albeit this system may be -and in most cases is- physically distributed, it remains as a logically centralized point of control as shown in Fig. 1. We propose to keep centralized the service/functions catalogs and the general NFV orchestration. The management of the virtual resources other than network (storage, computing) is also partially centralized. However, we advocate that the management of the virtual network can be completely offloaded to the infrastructure. Thanks to an enhanced SDN infrastructure it can be totally decentralized and auto-coordinated. Similarly, the VNF management can be also partially offloaded. The creation and destruction of VNF instances is still coordinated by a centralized entity (e.g. OpenStack.org). However, the monitoring, balancing and load assignment is decentralized and coordinated directly by the enhanced SDN infrastructure.

b) SDN infrastructure enhancement: The MANO system comprises different instances of an SDN controller to control the network. To achieve an enhanced SDN infrastructure, they must be isolated and pushed close to the data-plane devices they control. Furthermore, part of the MANO system itself should be partially distributed over those controller instances to achieve better local control, as shown in Fig. 2. Previous works already discuss collocating NFV and SDN elements with the data-plane nodes [8]. We seek to also effectively offload the control to those elements. The state database remains as part of the centralized MANO, but it is mostly updated by the decentralized controllers. Controllers collocated with the data-plane devices have richer information about the traffic. They can make faster and better decisions than a centralized MANO.

c) Offload redundant VNF functionality: This enhanced SDN infrastructure with decentralized MANO modules offers a general framework where different VNF types can be allocated. Features common among the VNFs (e.g. resilience, load balancing, etc) can be offloaded to the local MANO modules that use the federated global state to coordinate (see Fig. 2). This results in modular, optimized and compact VNFs, which enables a VNF-agnostic architecture that can efficiently handle the different kinds of traffic expected on ISP networks.

d) Overlay encapsulation: An overlay encapsulating traffic over the legacy infrastructure can overcome the constraint of the network equipment already in place. We differentiate three parts in the network, *overlay, underlay and outerlay* (as shown in Fig. 3). Overlay is the virtual network instantiated by the architecture through encapsulation. Underlay is the legacy network beneath based on off-the-shelf hardware. Outerlay are the external networks that generate/receive the traffic and that connect to the overlay through enhanced SDN edge-nodes.

e) Strong identity-location decoupling: To support the model of an VNF-agnostic overlay-based system with decentralized control, the architecture must enforce a strong decoupling of identity and location semantics and introduce different levels of indirection. We aim to solve NFV challenges by moving the network appliances to the datacenter and then getting the outerlay traffic there. Identity-location split is required to maintain real time mappings of VNFs to datacenter servers, overlay traffic to underlay tunnels and outerlay clients to offered services.

V. Architecture

Building on the design principles discussed in section IV, we propose the NFV architecture depicted in Fig. 3. To illustrate the architecture, let us assume that an operator has deployed a service to enhance HTTP traffic on its network. This service leverages on using a firewall, to check that the subscriber is not accessing a malicious site, and then using a TCP optimizer to boost the transmission performance. In the figure, the VNFs hosted in Datacenter 1 implement firewall functionality while those hosted in Datacenter 2 are TCP optimizers. In Fig. 3 a HTTP flow from a subscriber arrives to the edge-node on the left which detects that the flow is HTTP traffic subject to be enhanced. The edge-node queries the global state database to find the VNF chain assigned to that particular flow. Since no chain has been computed, it creates one itself. Based on the current state of the system stored in the database, the edge-node decides that the traffic will go through VNF-1 (firewall) and then through VNF-4 (TCP optimizer). This decision is made publicly available by storing the flow-to-VNF-chain affinity in the global state database. Using this global information, the rest of the edge-nodes can properly forward the traffic. First across the firewall, then through the TCP optimizer, and finally to the Internet. The rest of this section contains more details on the architecture.

A. Edge-nodes

Due to the legacy equipment already in place it is not cost-effective to upgrade all nodes in the network to support the required NFV capabilities. Therefore, the architecture relies on just upgrading the nodes at the network edges i.e. the ingress/egress points for clients' networks and the locations

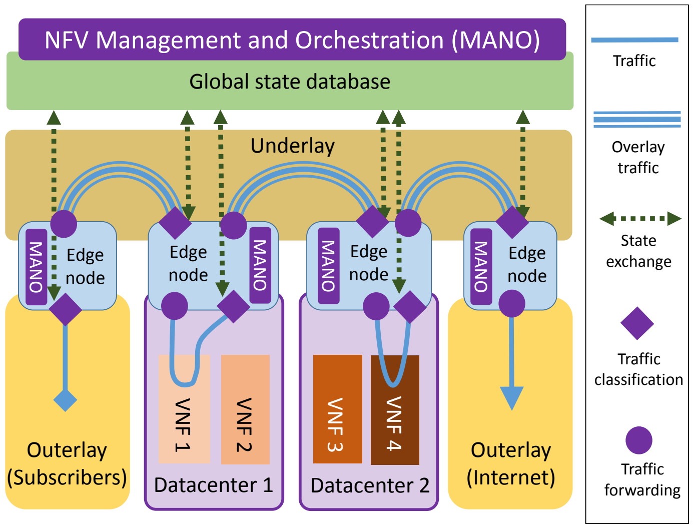

Fig. 3. **Proposed architecture**

where the VNFs are hosted. For the VNFs case, these edge-nodes may be the switches at the top of virtualization racks and/or the gateways of the datacenters hosting the VNFs. Given the characteristics of ISP networks, these edge-nodes should offer flow granularity for packet processing while keeping line-rate throughput on the data-plane and low-latency times for the control-plane.

To achieve this, we propose the edge node design depicted in Fig. 4. An SDN controller is collocated with a hardware SDN switch to minimize switch-controller latencies. This hardware SDN switch is able to process the traffic at flow granularity and line-rate speed via minimizing the lookup time i.e. only performing exact match lookup over a minimal set of packet fields (e.g. 3-tuple). Any packet that does not hit an exact match entry (i.e. no rule available for its flow) is sent upwards to a software SDN switch. Although slower, the software SDN switch allows performing more granular (e.g. 5-tuple) flow look-ups and define as many rules as needed. In general, to classify a packet more fields are needed than to forward it. The software SDN switch contains detailed rules to classify the flow and find the appropriate MANO service module within the SDN controller.

These MANO modules are per-service (e.g. HTTP enhancement) specific software pieces that can assign flows to VNF chains and program accordingly the hardware switch to forward them. Once the software switch hands the flow to the proper MANO module, the module checks if there is already cached an VNF chain suitable for the flow. If that is not the case, it uses the controller's database interface to retrieve a suitable chain from the federated information (see section V-B). If no suitable chain exists for that specific flow, the service module has to compute one itself as described in section V-D. After retrieving/computing the VNF chain, the MANO service module uses the controller's southbound interface to program (e.g. via OpenFlow [2]) the exact match rules in the hardware switch. Subsequent packets of the flow will hit the exact match entry and be processed at hardware level.

B. Federated global state

A physically distributed, but logically centralized, database federates all the state generated locally at the edge-nodes (e.g. computed VNF chains, etc). This database makes the state globally available to the whole infrastructure. It also stores general MANO information (network services catalog, VNF catalog, VNF instances, infrastructure status, etc) [7]. A summary of the information stored is provided below.

- *VNF class* $\rightarrow$ *VNF instances*: The abstract VNF classes that the different service use are instantiated into (and mapped to) specific VNF instances.
- *Flow* $\rightarrow$ *VNF chain*: Each flow already processed is mapped to its assigned chain of VNF instances.
- *VNF instance* $\rightarrow$ *Instance status*: Per each VNF instance the database stores i.e its current location, the number of flows assigned to it, etc.

The database follows a strong location-identity decoupling model to store the information which allows to easily introduce different levels of indirection. This entitles end-points to smoothly move across different access networks and allows VNFs to be elastically allocated both inside and outside a datacenter. For instance, in Fig. 3, VNF 1 (i.e. identity) can be seamlessly migrated from Datacenter 1 to Datacenter 2 (i.e. location) following this schema.

For the database implementation, the architecture uses a Distributed Hash Table (DHT) database back-end. Such databases use hashes to index the information and thus offer a scalable storage with a delimited query time. In terms of available solutions, Cassandra [9] can fulfill the requirements due to its good availability and excellent scale-out capacity [10]. For the front-end interface to the database, the architecture uses LISP [11], [12], a pull-based protocol that allows retrieving identity-to-location mappings from a central repository. LISP fits well the identity-location split model required by the architecture and offers an interoperable (i.e. IETF-baked) and lightweight mechanism to retrieve state.

Given that the large size of the network leads to considerable state to store, to keep the architecture scalable the state is only pulled on demand by edge-nodes. Therefore, in order to notify changes and keep the state consistent the database follows a publish-subscribe mechanism [13]. As an example, if in Fig. 3 VNF 1 has to be moved to Datacenter 2, the edge-node at the subscribers network will be notified and start encapsulating the flow towards Datacenter 2 instead of Datacenter 1.

C. VNFs

The VNFs are allocated in generic virtualization racks at different datacenters with an edge-node as ToR switch or datacenter gateway. Due to the encapsulation and the location-identity split, the VNFs can be dynamically moved across hosts, racks or datacenters. Therefore, the model allows to both encapsulate the traffic towards where the VNFs are and/or move the VNFs close to where the traffic is. All this path

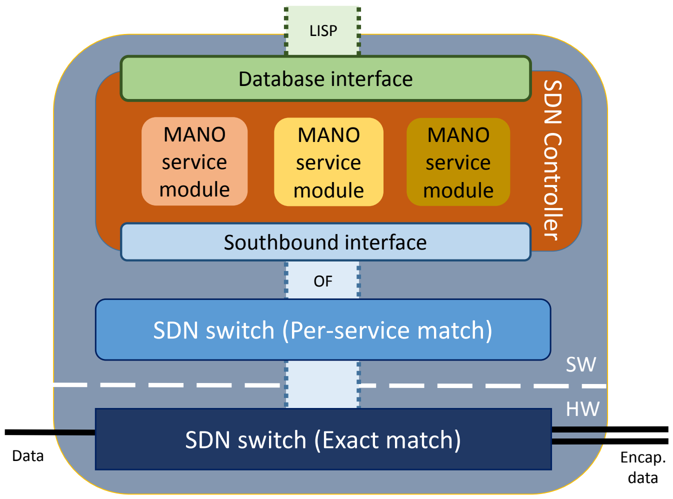

Fig. 4. **Edge node internals**

computation complexity is offloaded to the MANO service modules at the edge nodes.

In this architecture, the VNFs are unaware of the rest of the system (i.e. they do not know the next hop for a flow). Therefore, the scope of the VNF state is restricted to flow processing. This simplifies the elastic allocation of VNFs and the deployment of new services, since different VNFs from different services can be easily chained.

Ideally, each VNF should perform only one single task and complex services should be created by chaining different individual VNFs. This enables a flexible system that can scale-out in a modular fashion. For instance, if a VNF is experiencing high load, that specific VNF can be scaled-out independently without affecting other VNFs in the chain.

In this sense, the architecture trims out redundant logic common to all VNFs and moves it to the distributed MANO modules. Scalability, load balancing, high availability, etc, are decoupled from the VNFs and offloaded to the infrastructure. As an example, a firewall VNF processes packets unaware of any balancing policies. Its local MANO module monitors it and takes care of reassign flows to properly balance the load among similar firewall VNFs.

D. Management and orchestration (MANO)

The architecture is oriented towards deploying services (e.g. HTTP enhancement) via decentralized MANO modules. The central MANO system installs these service-specific modules in the edge node and programs the software switches to forward the traffic to them. Each service defines the type of traffic to be processed (e.g. HTTP) and the VNF classes to apply (e.g. firewall, TCP optimizer). The central MANO system is in charge of instantiating the VNFs for the service. The decentralized MANO modules build on-demand VNF chains based on available VNF instances, drive the traffic through the VNF chain and notify the central MANO system when a VNF needs to be migrated.

The service description defines the classes of VNFs to chain, but the service modules decide on real time which is the best VNF chain among all possible VNF instances. For instance, for a real-time analytics service the best chain may be composed of VNF instances placed in low-latency locations while for an on-the-fly video decoding service the chain may comprise the currently less loaded VNF instances. A computed VNF chain is stored in the database to make it available globally and cached locally to assign it to similar flows in the future. The distributed service-specific MANO modules monitor the traffic and the VNFs and are synchronized with the federated global state and with the central MANO subsystem. Therefore, they can reassign flows to different chains or recompute chains if required.

VI. Qualitative analysis

A. Advantages

An NFV architecture leveraging on SDN comprises several benefits. First, a flexible and rich control of the network thanks to the SDN controllers. Second, inherent support for traffic engineering and load balancing enabled by the SDN fabric. Furthermore, the decentralized NFV architecture that we propose presents a set of additional advantages.

1) Decentralization boosts scale-out: Since the coordination required among the different parts of the architecture is relaxed, it is easier for these parts to scale-out independently. This can be achieved for the architecture as a whole (e.g. adding more edge nodes) or for each component individually (e.g. adding more physical servers to an edge node cluster).

2) Flow granularity even at large networks: The optimized flow lookup allows for more flows to be handled per each hardware switch and thus reduces the cost of scaling-out the edge nodes to allocate more traffic. In general, all architecture components are designed to keep flow granularity despite the network size. Edge nodes process flows in parallel independently, VNFs keep only per-flow state and the federated database uses a plain namespace with constant access time.

3) Better per-flow decisions: The decisions on how to process a flow are taken close to the data-plane devices carrying the flow itself. Therefore, more and richer per-flow information is available. The flow granularity processing and this detailed per-flow information entitle for complex per-flow decisions, something that is challenging to accomplish with traditional logically centralized architectures.

4) VNF outsourcing: The combination of an architecture that is VNF-agnostic and of VNFs that are simple, light and interoperable, enables VNF outsourcing. The VNFs do not need to be specifically developed for the particular NFV system but rather can be developed by third parties and smoothly integrated with other VNFs. The architecture eases the development of such outsourced VNFs, since VNF-vendors can leverage on the mechanisms offered by the infrastructure and thus avoid dealing with ISP networks scalability or availability requirements.

B. Challenges

1) Global state query latency and extra singling: Relaying on a global state database presents some challenges. First, the state retrieval imposes an inherent latency on the operation of edge-nodes. Nevertheless, technologies already available should offer low enough query time. For instance, Cassandra queries take only a few milliseconds even under high loads [10] and an optical underlying transport induces latency in the order of microseconds [14]. Second, the dependency on the global state may result in an overhead of message exchanges. However, caching techniques at the edge-nodes and careful design of service requirements may render this extra signaling overhead negligible.

2) State inconsistencies on the decentralized system: The decentralized nature of the architecture may introduce state inconsistencies. However, the state update mechanisms in place make the state to eventually converge. Furthermore, local controllers can adapt their control policies to face these temporary state divergences. As an example, the edge-node for a particular firewall VNF decides to migrate it to a less loaded location. As a consequence, this new location is published in the database and notified to edge-nodes that previously retrieved the firewall's location. While the update propagates through the system, the edge-node at the old location will redirect incoming packets address to the firewall towards the new location.

3) Lack of control for the underlay network: The architecture uses the underlay network and has to rely on its correct operation. If that is not the case the architecture has no control over it and is unable to fix the problem. However, ISP networks will likely have their own troubleshooting and healing mechanisms, as assumed in [5]. Furthermore, in the case of a major connectivity problem, the enhanced SDN infrastructure may be able to transparently detour the traffic thanks to the identity-location split schema enforced.

4) Edge-node implementation: The proposed service-based decentralization results in complex edge-nodes that need to remain scalable. On one hand, each local MANO module is independent of the others and its performance is not affected by the number or complexity of other modules. Therefore, scale-out requirements can be met with a cluster-friendly controller (e.g. OpenDaylight.org) able to distribute the load. On the other hand, the hardware switch is agnostic to the service complexity or its number since it only considers independent exact match rules. Thus, it can be scaled-out across several hardware devices.

The bottleneck of the system is the software switch. In this case, contrary to the rules allocated in the hardware switch, the rules required to support more services or more complex ones comprise wildcarded fields, longest prefix match lookups and different priorities (since these rules will likely overlap). This makes the complexity of flow classification at the software switch to increase non-linearly with respect to the complexity or number of services. However, the architecture needs a software switch capable of achieving the linear scalability required by the large number of flows expected, despite the flow heterogeneity and the non-linear complexity faced in the flow classification.

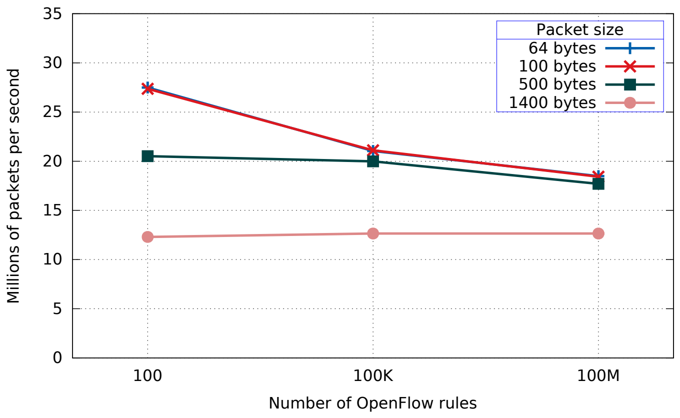

Fig. 5. **In-house software switch performance with millions of rules**

VII. Software switch implementation

From the analysis on section VI, we conclude that the scalability on the system will be capped out by the performance achieved by the software switch. To cope with the requirements of the ISP scenario, the software switch must be able to keep a high packet throughput despite the number of rules and the heterogeneity of the traffic.

We measured the performance of currently available software switches, particularly Open vSwitch (openvswitch.org), and we were able to achieve 11M packets per second (pps) using Open vSwitch 2.3.1 DPDK-optimized (dpdk.org) on a single core with 100 OpenFlow rules and less than 100 traffic flows. This number is similar to the one reported in [15] and, at the best of our knowledge, this is due to caching lookup results for known flows to effectively bypassing the OpenFlow lookup tables. When we raised the number of flows to 500K, a number closer to the ISP scenario, the performance dropped to 300K pps. We observed also non-linear scaling since a 8-core configuration only achieved 1.2M pps. The hardware used for these tests was similar to the one described later in this section.

To achieve ISP performance requirements we implemented our own in-house software switch, written in C and leveraging on DPDK. It is based in a multithreaded design where all threads have access to the rules from a common memory space. Each thread handles a subset of the flows distributed to it based on 5-tuple hashing performed by the NICs, using Receive Side Scaling (RSS) technology with a queue per thread. The OpenFlow tables are presented as static tables and updates are performed on a shadow copy of those tables. Periodically the shadow copy is switched with the active table set, updated with the changes made on the shadow copy and from there updates commence on the new shadow. The key to the high performance implementation is that for every packet, the relevant rules are fetched into cache memory just in time for lookup. By pipelining the rule prefetching (i.e. handling a few packets in parallel by each thread) the throughput is achieved by effectively always referencing rules that reside in the CPU cache (and not in off-chip memory). As a result, the performance is almost independent of the number of rules.

To measure the scalability of the proposed software switch we performed the following benchmark. We ran the switch on an Intel-based server with dual Intel Xeon E5-2690 2.9GHz 8-core per socket CPU (i.e. 16 total cores) with 128GB of RAM and a set of 16 interfaces of 10Gbps each. We populated the switch with rules ranged from 100 to 100M and we generated traffic evenly distributed across all rules (i.e. the traffic was forged to hit all rules at the same rate). Fig. 5 shows the packets per second processed by the switch for different numbers of rules and packet sizes. In all cases the delay per packet was constant and around 50 μs. The figure shows how the switch scales almost linearly and achieves the requirements of the architecture.

VIII. Conclusion

The architecture presented in this paper addresses the challenges of NFV for ISP networks via partially decentralizing the MANO system. Contrary to most NFV proposals, the SDN controllers used by the MANO are collocated with their controlees and provisioned with local MANO modules. This enables faster local processing by means of reducing centralization. The architecture proposes a good tradeoff of complexity, performance and scalability by decentralizing some components while keeping a centralized state.

Acknowledgments

This work has been partially supported by a Cisco research grant, by the Spanish Ministry of Education under grant FPU2012/01137, by the Spanish Ministry of Economy and Competitiveness under grant TEC2014-59583-C2-2-R and by the Catalan Government under grant 2014SGR-1427.

References

[1] B. Han, et al., ”Network function virtualization: Challenges and opportunities for innovations,” in *IEEE Communications Magazine*, vol. 53, no. 2, pp. 90-97, Feb. 2015.

[2] D. Kreutz, et al., ”Software-Defined Networking: A Comprehensive Survey,” in *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14-76, Jan. 2015.

[3] H. Hawilo, et al., ”NFV: state of the art, challenges, and implementation in next generation mobile networks (vEPC),” in *IEEE Network*, vol. 28, no. 6, pp. 18-26, Nov.-Dec. 2014.

[4] P. Berde, et al., ”ONOS: towards an open, distributed SDN OS,” in *Proceedings of the third workshop on Hot topics in software defined networking (HotSDN ’14)*, ACM, New York, NY, USA, pp. 1-6, 2014.

[5] S. Jain, et al., ”B4: experience with a globally-deployed software defined wan,” in *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM (SIGCOMM ’13)*, ACM, pp. 3-14, 2013.

[6] ”Network Functions Virtualization (NFV); Architectural Framework,” ETSI Group Specification GS NFV 002 (v. 1.2.1), Dec. 2014

[7] ”Network Functions Virtualisation (NFV); Management and Orchestration”, ETSI Group Specificaion GS NFV-MAN 001 (v. 1.1.1), Dec. 2014.

[8] J. Matias, et al., ”Toward an SDN-enabled NFV architecture,” in *IEEE Communications Magazine*, vol. 53, no. 4, pp. 187-193, April 2015.

[9] A. Lakshman and P. Malik. ”Cassandra: a decentralized structured storage system.” in *ACM SIGOPS Operating Systems Review*, vol. 44, no. 2, pp. 35-40, April 2010.

[10] T. Rabl, et al. ”Solving big data challenges for enterprise application performance management.” in *Proceedings of the VLDB Endowment*, vol. 5, no. 12, pp. 1724-1735, August 2012.

[11] D. Farinacci, et al. ”The locator/ID separation protocol (LISP).”, RFC 6830, IETF, Jan. 2013.

[12] A. Rodriguez-Natal et al., ”LISP: a southbound SDN protocol?,” in *IEEE Communications Magazine*, vol. 53, no. 7, pp. 201-207, July 2015.

[13] S. Barkai, et al. ”LISP Based FlowMapping for Scaling NFV.”, draft-barkai-lisp-nfv-08, Internet-Draft, IETF, June 2016, work in progress.

[14] C. Kachris and I. Tomkos, ”A Survey on Optical Interconnects for Data Centers,” in *IEEE Communications Surveys & Tutorials*, vol. 14, no. 4, pp. 1021-1036, Fourth Quarter 2012.

[15] P. Emmerich, et al., ”Performance characteristics of virtual switching,” in *IEEE 3rd International Conference on Cloud Networking (CloudNet)*, Luxembourg, pp. 120-125, 2014.

Alberto Rodriguez-Natal received a BSc (2010) in Computer Science from the University of Leon (Spain) and a MSc (2012) and PhD (2016) from the Technical University of Catalonia (Spain). He has also been a visiting researcher (2014) at the National Institute of Informatics (Japan). He joined Cisco in 2016 where he continues his research on new network architectures with special focus on Software-Defined Networking and programmable overlay networks.

Vina Ermagan is a Sr. Technical Leader in the Chief Technology and Architecture Office at Cisco Systems. She joined Cisco in 2008, and has been working on research, design, and development of network virtualization and SDN technologies ever since. She has initiated projects to implement LISP in Open vSwitch (OVS), OpenDaylight, and FD.io . Vina received her MSc in Computer Science from UC San Diego in 2008, and her BSc in Computer Engineering from Sharif University of Technology.

Ariel Noy is responsible for setting the HPE, CSB ConteXtream technical direction, architecture, and ongoing innovation and Proof of Concepts (PoCs). Ariel co-founded ConteXtream and served as CTO for 9 years. Prior to ConteXtream he was a technical leader at Cisco Systems. Ariel joined Cisco when it acquired Sheer Networks, where he was a co-founder and VP Engineering. Ariel has 11 patents and has a BSc in Math and Computer Science from University of Tel-Aviv (1996).

Ajay Sahai currently leads Open NFV partner solutions at Hewlett Packard Enterprise. Previously he was Director Technical Marketing at ConteXtream an SDN startup that was acquired by HPE in 2015. In his 20+ year career Ajay has focused on the wireless/networking/telecom arena and worked in both business and technical roles. Ajay has a BSEE and an MBA. He has a couple of issued patents and several applications pending.

Gideon Kaempfer is a Distinguished Technologist and Chief Architect of HPE Contextream where he has been innovating in the fields of SDN and NFV since 2008. He is a serial entrepreneur bringing with him 25 years of experience in Telecom networking, routing, hardware and software system design and development. Before joining HPE, Gideon held senior architect or CTO positions at Contextream, Xring Technologies, Axxana, Expand Networks, FlowInspect, SilverKite and Charlottes Web Networks. Gideon is a TSC member of the OpenDaylight open source SDN platform. He holds an MSc in Computer Science from the Technion - Israeli Institute of Technology.

Sharon Barkai is a Distinguished Technologist at Hewlett Packard where he specializes in network-databases and functional grids. Sharon holds 13 patents and founded grid startups, Sheer semantic networks (CSCO), Xeround sound netDB, ConteXtream map-assisted overlays (HP), and Fermi Cloud lambda-edge. Sharon received BA in CS Mathematics from Hebrew University, MSc. in CS from Columbia University School of Engineering and applied science.

Fabio Maino is a Distinguished Engineer at Cisco Systems, in the Chief of Technology and Architecture Office, where he leads a team that focuses on driving innovation on Network Virtualization and SDN. He has about 50 patents issued or filed with the US PTO, and has contributed to various standardization bodies including IEEE, IETF, and INCITS. Fabio has a Ph.D. in Computer and Network Security and a M.S. (”Laurea”) in Electronic Engineering from Politecnico di Torino, Italy.

Dr. Albert Cabellos is an associate professor at Universitat Politcnica de Cataulnya (Department of Computer Architecture), he has been visiting professor at Cisco Systems (Santa Jose, US), KTH (Kista, Stockholm), Agilent Technologies (Edinburgh, UK), Massachusetts Institute of Technology (Cambridge, USA) and UC Berkeley (Berkeley, USA). He has participated in several research projects funded by companies (Cisco, Intel, Samsung, etc) as well as publicly funded (FP7, H2020, NSF and national funding agency). Albert is the co-founder of the Open Overlay Router (http://openoverlayrouter.org) open-source project and the co-chair of the Proposed Network Machine Learning WG at Internet Research Task Force (IETF).