AINFV: Analysis of Isolation (memory/packet) in Network Function Virtualization Overview

Abdul Ahad Ayaz

Summary

Network Function Virtualization (NFV) is a new way of defining network with the help of software that way previously done with hardware middle-boxes. These days, Service Providers fully utilize the IT Virtualization Technologies (using commodity servers) for fast deployment of new network services with minimal cost. In 2012, ETSI[6] was formed that defined the requirements and standardization of NFV. There is performance overhead between NFV and hardware middle-boxes in terms of packet transfer rate, latency etc. Another main thing to consider is the isolation, there are two types: memory isolation and packet isolation. Memory isolation is memory used by one NF should not be accessible by another NF, this can be done by using VMs/Containers. Packet isolation is when NFs are chained, at any point in time only one NF should have the reference to that particular packet. Packet isolation is hard to achieve as compared to memory isolation in NFV as it involve the copying of the packets from one NF to other using vSwitch. These isolations ensures security but at the cost of performance. The performance metric used are packet processing (i.e. Million Packets per Second [MPPS]) and latency. The model paper NetBricks[14] framework ensure the above isolations without the use of multiple VMs/Containers and packet copying using vSwitch. All NFs are chained in a single VM and communicate using function call. It provides the same performance as without using any isolation. NetBricks provide the high-level abstractions using RUST[20] to developers for new developments and low-level optimization. Prior to NetBricks many advancements had been done to ensure the isolation, but the performance was not up to the mark for example: NetVM[8], xOMB[3], ClickOS[11], mSwitch[7], HyperSwitch[2]. NetVM is virtualization-based platform. It has shared memory that uses DPDK[4] for zero-copy delivery between VMs ensuring isolation. ClickOS uses the Click[9] as a main programming model for middle-boxes and creating minimal OS . It runs in para-virtualized environment. It ensures the isolation and low packet delay. HyperSwitch is virtualization-based platform, it implements the virtual switch inside the hypervisor for inter-VMs communication and ensure memory isolation. Most of the above mentioned developments used VMs/Containers i.e one NF per VM/Container for isolation. OpenNetVM[24] framework is similar to NetBricks and is based on NetVM architecture using Dockers Container. SafeBricks[16] is based on

NetBricks framework. It is used to protect the traffic processing in cloud by executing the NFs in *hardware enclaves*(i.e. Intel SGX[12]). HyperNF[23] framework is used to fully utilized the available resources(i.e. CPU cores) for processing. Packet processing is done using the hypercall instead of involving the vSwitch every time. HyperNF is based on standard NFV architecture whereas NetBricks rewrites the software middle-boxes. libVNF[13] is an open-source library that provide the high-level abstractions as NetBricks. It also enables clustered VNF implementations that are not supported by the NetBricks. YANFF[15] framework provides high-level abstraction using Go language. It uses the scheduler for packet processing over multiple cores, whereas in NetBricks it is done by shuffle calls. G-NET[25] is based on GPU virtualization, GPU scheduler is used to optimize the throughput. G-NET uses *IsoPointer* for data isolation.

References

[1] Linux Foundation. OPNFV. URL `https://www.opnfv.org/`.

[2] Hyper-Switch: A Scalable Software Virtual Switching Architecture. *Atc '13*, pages 13–24, 2013. URL `https://www.usenix.org/conference/atc13/technical-sessions/presentation/ram`.

[3] James W. Anderson, Ryan Braud, Rishi Kapoor, George Porter, and Amin Vahdat. xOMB: extensible open middleboxes with commodity servers. In *Proceedings of the eighth ACM/IEEE symposium on Architectures for networking and communications systems - ANCS '12*, page 49. ACM Press, 2012. URL `http://dl.acm.org/citation.cfm?doid=2396556.2396566`.

[4] Intel Corporation. Intel ® Data Plane Development Kit. (June), 2014. URL `https://www.dpdk.org/`.

[5] Jingpu Duan, Xiaodong Yi, Junjie Wang, Chuan Wu, and Franck Le. NetStar: A Future/Promise Framework for Asynchronous Network Functions. *IEEE Journal on Selected Areas in Communications*, 37(3):600–612, mar 2019. URL `https://ieeexplore.ieee.org/document/8635508/`.

[6] ETSI. Network Functions Virtualisation,An Introduction, Benefits, Enablers, Challenges & Call for Action. Technical Report 1, 2012. URL `http://portal.etsi.org/NFV/NFV_White_Paper.pdf`.

[7] Michio Honda, Felipe Huici, Giuseppe Lettieri, and Luigi Rizzo. mSwitch: a highly-scalable, modular software switch. In *Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research - SOSR '15*, pages 1–13. ACM Press, 2015. URL `http://dl.acm.org/citation.cfm?doid=2774993.2775065`.

[8] Jinho Hwang, K. K. Ramakrishnan, and Timothy Wood. NetVM: High performance and flexible networking using virtualization on commodity platforms. *IEEE*

Transactions on Network and Service Management, 12(1):34–47, mar 2015. URL http://ieeexplore.ieee.org/document/7036139/.

[9] Eddie Kohler, Robert Morris, Benjie Chen, John Jannotti, and M. Frans Kaashoek. The click modular router. *ACM Transactions on Computer Systems*, 18(3):263–297, aug 2000. URL http://portal.acm.org/citation.cfm?doid=354871.354874.

[10] Wei Mao, Zhen Shen, and Xiaolong Huang. Facilitating Network Functions Virtualization by Exploring Locality in Network Traffic. In *Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence - CSAI '18*, pages 495–499. ACM Press, 2019. URL http://dl.acm.org/citation.cfm?doid=3297156.3297247.

[11] Joao Martins, Mohamed Ahmed, Costin Raiciu, Vladimir Olteanu, Michio Honda, Roberto Bifulco, Felipe Huici, and Implementation Nsdi. ClickOS and the Art of Network Function Virtualization. pages 459–473, 2014. doi: 10.1007/978-3-642-38703-6_48. URL https://www.usenix.org/conference/nsdi14/technical-sessions/presentation/martins.

[12] Frank McKeen, Ilya Alexandrovich, Alex Berenzon, Carlos V. Rozas, Hisham Shafi, Vedvyas Shanbhogue, and Uday R. Savagaonkar. Innovative instructions and software model for isolated execution. pages 1–1, 2013. URL https://www.eit.lth.se/fileadmin/eit/courses/eitn50/Literature/hasp-2013-innovative-instructions-and-software-model-for-isolated-execution.pdf.

[13] Priyanka Naik, Akash Kanase, Trishal Patel, and Mythili Vutukuru. libVNF: A Framework for Building Scalable High Performance Virtual Network Functions. In *Proceedings of the 8th Asia-Pacific Workshop on Systems - APSys '17*, pages 212–224. ACM Press, 2017. URL http://dl.acm.org/citation.cfm?doid=3124680.3124728.

[14] Aurojit Panda, Sangjin Han, Keon Jang, Melvin Walls, Sylvia Ratnasamy, and Scott Shenker. NetBricks: Taking the V out of NFV. In *OSDI'16*, pages 203–216, 2016. URL https://www.usenix.org/conference/osdi16/technical-sessions/presentation/panda.

[15] Ilya Philippov and Areg Melik-Adamyan. Novel approach to network function development. In *Proceedings of the 13th Central & Eastern European Software Engineering Conference in Russia on - CEE-SECR '17*, pages 1–6. ACM Press, 2017. URL http://dl.acm.org/citation.cfm?doid=3166094.3166111.

[16] Rishabh Poddar, Chang Lan, Raluca Ada Popa, and Sylvia Ratnasamy. SafeBricks: Shielding Network Functions in the Cloud, 2018. URL https://www.usenix.org/conference/nsdi18/presentation/poddar.

[17] Vyas Sekar, Norbert Egi, Sylvia Ratnasamy, Michael K. Reiter, and Guangyu Shi. Design and Implementation of a Consolidated Middlebox Architecture. In *Proc. USENIX NSDI*, page 24, 2012. URL https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/sekar.

[18] Vyas Sekar, Norbert Egi, Sylvia Ratnasamy, Michael K. Reiter, and Guangyu Shi. Design and Implementation of a Consolidated Middlebox Architecture. In *Proc. USENIX NSDI*, page 24, 2012. URL https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/sekar.

[19] MN Thadani. An efficient zero-copy I/O framework for UNIX. Technical Report May, 1995. URL https://dl.acm.org/citation.cfm?id=974947.

[20] The Rust Team. Rust Programming Language, 2016. URL https://www.rust-lang.org/.

[21] Wenfei Wu and Ying Zhang. Network function modeling and its applications. *IEEE Internet Computing*, 21(4):82–86, 2017. URL http://ieeexplore.ieee.org/document/7994546/.

[22] Hamid Yaghoubi, Nariman Barazi, and Mohammad Reza. Maglev: A Fast and Reliable Software Network Load Balancer . In *Infrastructure Design, Signalling and Security in Railway*, pages 523–535. 2012. URL https://www.usenix.org/conference/nsdi16/technical-sessions/presentation/eisenbud.

[23] Kenichi Yasukata, Felipe Huici, Vincenzo Maffione, Giuseppe Lettieri, and Michio Honda. HyperNF: building a high performance, high utilization and fair NFV platform. In *Proceedings of the 2017 Symposium on Cloud Computing - SoCC '17*, pages 157–169. ACM Press, 2017. URL http://dl.acm.org/citation.cfm?doid=3127479.3127489.

[24] Mykola Yurchenko, Patrick Cody, Aaron Coplan, Riley Kennedy, Timothy Wood, and K. K. Ramakrishnan. OpenNetVM. In *Proceedings of the 2016 workshop on Hot topics in Middleboxes and Network Function Virtualization*, pages 1–2. ACM, 2018. URL https://dl.acm.org/citation.cfm?id=2940155.

[25] Kai Zhang, Bingsheng He, Zeke Wang, Bei Hua, Jiayi Meng, Systems Design, and Implementation Nsdi. G-NET : Effective GPU Sharing in NFV Systems. 2018. URL https://www.usenix.org/conference/nsdi18/presentation/zhang-kai.