Towards Multi-layer Resource Management in Cloud Networking and NFV Infrastructures

Kurt Tutschku, Vida Ahmadi Mehri, and Anders Carlsson
Blekinge Tekniska Högskola,
Department of Telecommunication Systems,
371 79 Karlskrona, Sweden.
Email: [kurt.tutschku|vida.ahmadi.mehri|anders.carlsson]@bth.se

Abstract: Cloud Networking (CN) and related concepts offer appealing novelties to Cloud Computing (CC) customers. They can do a one-stop-shopping for network-enhanced cloud services. In addition, the costs of such services might be low due to multiple customers sharing the infrastructures. Moreover, telecommunication network operators are adopting the CN in their *Network Functions Virtualisation (NFV)* framework for reducing costs and increasing the flexibility of their networks. The technical appeal of CN comes from the tight integration of CC and smart networks. The economical attractiveness results from avoiding dedicated hardware, sharing of resources, and simplified resource management (RM) as seen by the users respectively by the applications. The vision of cheap and integrated CN services is obviously attractive, but it is also evident that it will require more complex RM procedures for efficiently balancing the usage of all resources.

In this contribution, we suggest an initial architecture for integrated and practical RM in CN and NFV systems. The RM concept aims at locating and analysing performance bottlenecks, efficiency problems, and eventually discover unused resources. The suggested architecture is based on a layered view on the system. Moreover, we detail difficulties in practical resources usage monitoring which, in turn, define requirements for a RM architecture. The requirement analysis is based on measurements in a CN infrastructure.

I. Introduction

Cloud Networking (CN) [2] or related Network-as-a-Service (NaaS) concepts [6] offer appealing novelties to Cloud Computing (CC) customers. They can buy at the same time smart computing and networking, i.e. they can do a one-stop-shopping for sophisticated and network-enhanced cloud services. The costs of CN services might be low due to multiple customers sharing infrastructures. Moreover, telecommunication network operators are increasingly adopting CC concepts to implement *Network Functions Virtualisation (NFV)* [8] for reducing costs and increasing the flexibility of their networks.

These CN features form pronounced business opportunities for new CC providers, like CityNetwork Webhotell AB [5]. They can provide fine-tuned computing and networking infrastructure services at a competitive price to specific markets, such as small and medium-sized enterprises or government agencies. Moreover, these companies can serve in the future as the infrastructure service providers for telecommunication operators.

The technical appeal of CN solutions comes from the tight integration of cloud applications and smart networks, e.g by integrated content computation and distribution. The economical attractiveness results from avoiding the use of dedicated networks and equipment, the sharing of technical resources, and a simplified management for the users.

The vision of cheap, integrated and on-demand CN services are obviously attractive for customers and providers. However, the different nature of the shared resources (e.g. computation, storage or networking), the heterogeneity of the applied technologies (e.g different operating systems) or the different scope and operational requirements (e.g. distributed resources administered by different operators) make it evident that CN system might require a more complex operational procedure than each resource alone. The difficulties are expected to originate from the coordination of the applied resource sharing and provisioning mechanisms. For example, an increased sharing efficiency in the compute subsystem might require increased network communication. Enhanced network traffic in turn, however, might reduce network efficiency.

The resource sharing in CN system will largely be based on *virtualisation concepts*, such as *Network Virtualisation* [1], [4] or *Hardware Virtualisation* [15]. The detailed efficiency of virtualisation technologies is still under investigation [9]. Though, a concept that aims at resource management (RM) in CN needs to consider the efficiency of the individual technologies as well as of the relationships among them.

This contribution suggests an initial architecture for an integrated and practical resource management concept for CN system. The concept aims at enabling CN operators to locate and analyse performance bottleneck, efficiency problems, and eventually discover unused resources in their infrastructures. The proposed architecture is based on a layered view on the system and therefore denoted as a *multi-layer resource management* concept. Moreover, we will discuss practical difficulties in resource usage monitoring in CN infrastructures. These adversities are uncovered by measurements in an infrastructure that is close to the one offered by City Network. The discovered difficulties are influencing the design of suggested RM architecture.

This paper is structured as following: first, Sec. II discusses experiments, observations, and capabilities on practical resource monitoring. Sec. III provides a taxonomy of factors that influence RM in CN infrastructures. Sec. IV outlines the suggested resource management architecture. Finally, Sec. V

provides a conclusion and an outlook on future work.

II. Experiments and Observations on Practical Resource Monitoring

We start the investigation of RM in CN with analysis of the practicalities and difficulties of resource monitoring on cloud servers. We focus initially on this hardware since we assume that they are the most novel addition by this networking paradigm. For the analysis, we conducted measurements on servers that are similar in their hardware and virtualisation software to the ones used in data centres (DCs) of the City Network (see Sec. II-D for hard- and software specifications).

A. Considered RM Scenario

The considered RM scenario is depicted in Figure 1. It consists of multiple users and a single CN operator. The operator offers the execution of virtual machines (VMs) that implement virtual appliances performing virtual functions (VFs). In virtualisation environments, the VMs are typically denoted as "guests" and the servers as "hosts". The operator offers the users a certain performance, either on VM level or on application/VF level.

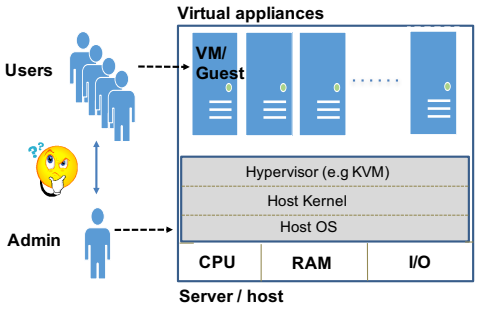

Fig. 1: Resource management scenario

The operator aims at loading the server with as many VMs as possible without scarifying the promised performance. The user aims at high performance for VFs at lowest cost, i.e. at the minimum of resources paid for. We focus in our measurements on CPU utilisation in order to confine the complexity. However, similar investigations might be necessary for understanding the other resources.

If a user experiences a less-than-agreed performance for a VM (or VF) then the operator might be contacted and asked for re-establishing the negotiated performance. Hence, the operator needs to be able to validate and judge the user's claims as well as to find a way the infrastructure can provide the promised resources.

Since resources might be used on small time scales, e.g. the run time of VFs is small, the RM must also perform on such small time intervals.

B. CPU Utilisation as a Resource Metric

We used *CPU utilisation ratio* as a metric for *usage* of the server. Its value is reported as a percentage. For purpose of simplification, we call this metric simply as *CPU utilisation*. It can be calculated from the amount of accumulated time a CPU is busy for handling work during the observation time interval. The accumulated value can be calculated from the inverse of the time a CPU is idle, or the time a CPU is running a user-level application or the system-level threads for each processor are served. Almost every CPU monitoring tool applies its own calculation for utilisation and their observation may not necessarily be comparable, cf. Sec. II-G3.

C. Questions on Practicality of Resource Monitoring

With regard to the above outlined scenario, we defined fundamental questions on resource monitoring and management for hosts and guests on CN servers:

1) *Load Scaling*: Is there a strong positive correlation of load and utilisation measured at the guest and the host? We denote this characteristic as *load scaling*. This scaling features enables the server administrator understand the relationship of guest and host load and predict the load on each other in a practical manner, i.e. in a bijective way.
   How strong do the utilisation values vary when they are observed during sampling intervals of different length?
2) *Dependency on Host OS*: Does the scaling behaviour depend on host operating systems?
3) *Dependency on Monitoring Tools*: Are there significant differences between the monitoring tools?

D. Experiment Configuration

The experiments used a server with an Intel® Xeon® EC-1230v2 (3.30GHz, 8 cores) CPU, 8GB RAM, and a 500GB SAS hard disk. It executed `kvm` [12] under `Ubuntu 12.04 LTS Desktop` [3] or `CentOS 6.6 Desktop` [14]. Each guest was allocated two virtual CPUs, 2GB of virtual RAM, and a 30GB virtual disk. This assignment is based on the KVM provisioning model that is based on CPU cores. The guest used `Ubuntu 14.04 LTS Server`.

E. Load Generation and Experimentation Parameters

Throughout the experiments, we increased utilisation in the guest by the `stress-ng -c N -l P -t 300` command [10] in 10% steps from 10% to 100% (`-l P` option). The number of loaded cores (`-c N`) was varied between one and two. Each utilisation level was iterated and measured at least 50 times and the load was imposed for 5min (`-t 300`) in order to have a relatively small monitoring time scale.

F. Monitoring Tools

top: We used the `top` [13] to monitor utilisation, which is the standard tool in `Linux` distributions for real-time performance view. The used command is: `top -b -d T1 -n T2`. The option `-b` starts a batch for monitoring and it is used for sending output to a file. The value `T1` provides the time interval between two utilisation samples and was set to 1sec. The parameter `T2` specifies the number of samples and was set to 300. We consider `top`'s the 1-min. average.

MPstat: The tool was used as an alternative for monitoring utilisation. It is part of the `sysstat` utilities [11]. The used command is: `mpstat -P ALL T1 T2`. The option `-P ALL` specifies that all processors will be monitored. The value `T1` defines the time between two measurements, i.e. 1sec. The

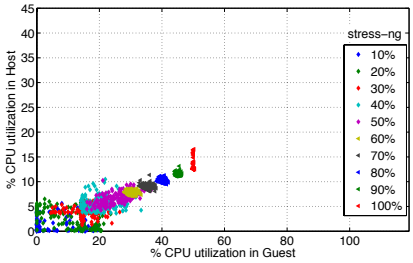

Fig. 2: Relationship for utilisation; one loading process on guest; host: Ubuntu; tool: mpstat

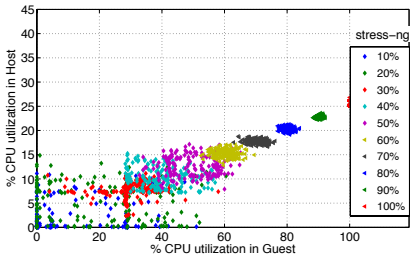

Fig. 3: Relationship for utilisation; two loading processes on guest; host: Ubuntu; tool: mpstat

parameter `T2` specifies the duration of the measurement. i.e. 300sec. We used $(100 - \texttt{\%idle})$ in our monitoring.

G. Measurements

1) Predictability of Load Scaling: The predictability of the scaling of the load and utilisation was observed using `mpstat` and is depicted in Figures 2 and 3. Figure 2 shows if the VM is loaded with 50% (i.e. 100% load from one process; i.e. using the option `-c 1` in `stress-ng`) then the host is loaded at about 12%. This behaviour can be expected for an eight core host and a VM with two vCPUs. However, the figures also reveal that the observed utilisations are not deterministic for all load levels and the measurements vary strongly. Hence, we conclude that there is predictable scaling, i.e. positive correlation, but this is weak and even weakens when the load on the guest is reduced. Hence, we conclude that the prediction accuracy decreases significantly at lower guest load.

Moreover, the measurements show that monitoring of utilisation on small time-scales are possible only as long as the VM is highly loaded. Moreover, Figure 3 reveals an even higher variation of the measured utilisation is observed when load is imposed on both vCPUs (i.e. `-c 2` in `stress-ng`). This behaviour indicates that the monitoring uncertainty is increased when stressing multiple vCPUs. This observation shows that the accuracy depends not only on the guest load level, but also on the type of guest load.

2) Dependency on Host OS: The dependency on the host operating system can be analysed by comparing Figures 3 and 4. Figure 4 reveals that `CentOS` shows typically a higher variation in the host at higher utilisation values than `Ubuntu`. Although `CentOS` is typically considered as a better choice as a host operating system, our measurements show that it

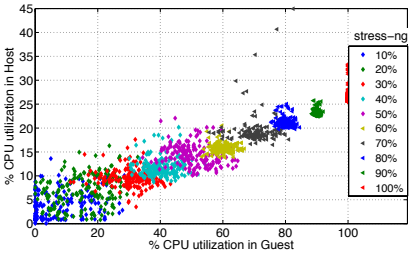

Fig. 4: Relationship for utilisation; two loading processes on guest; host: CentOS; tool: mpstat

exhibits a higher host variation and in turn a lower accuracy when describing utilisation. Therefore, `Ubuntu` seems to be a better choice when it comes to load optimisation.

3) Differences Between Tools: The comparison of Figures 2 and 3 for utilisation indicates that there might be an advantage for the `top` tool due to lower visual variations. However, the comparison of the tools requires further research.

4) Summary of Observations: The measurements reveal that resource monitoring is possible on small time scales, but an RM architecture has to consider the different monitoring and RM concepts of the tools and the operating systems.

III. Taxonomy of Cloud Networking Resources

As indicated above, RM in CN needs to consider the different types of resources and their relationships. Our approach for understanding them is based on the application of the well-known *Separation of Concerns (SoC)* design concept for computer programs [7] on CN systems. Figure 5 shows a first

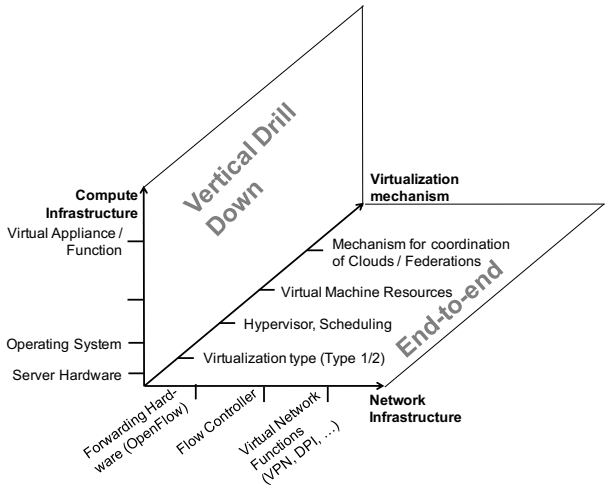

Fig. 5: Taxonomy of CN Resources

taxonomy of influence factors. It depicts a three-dimensional resource space that describes the scope of the resources. These resources might become performance bottlenecks. The dimensions in the description are selected according to the three major categories of infrastructure in CN: *a)* compute infrastructure, *b)* network infrastructure and *c)* virtualisation mechanisms.

The taxonomy indicates that an RM architecture requires the ability to a *vertical drill down*, i.e. multi-layer view of

the resources along the compute infrastructure and virtualisation mechanisms dimensions. In addition, the dimensions of network infrastructure and virtualisation mechanisms calls for a horizontal, end-to-end view. The knowledge about the efficiency of resource sharing is the key for RM in CN.

IV. A Multi-layer Resource Management Architecture

An initial framework architecture for multi-layer RM is shown in Figure 6. The figure details in the center the *Data*

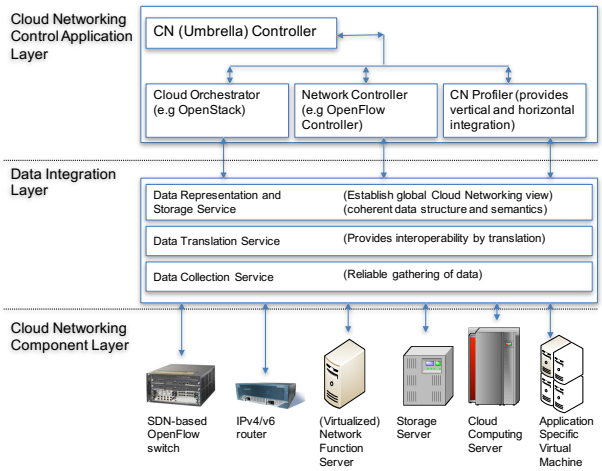

Fig. 6: Initial Multi-layer RM Architecture

Integration Layer (DIL), which provides a global and an interoperable view on the performance and efficiency of the CN infrastructure. It provides *a)* Data Representation and Storage Services for efficiency and load data; *b)* Data Translation Services that convert raw performance and efficiency data into interoperable data or statistics of multiple time scales; and *c)* Data Collection Services to reliable, collect performance and efficiency values. In detail, the DIL will comprise in the Data Representation and Storage Service, cf. Figure 7, data objects with interoperable semantics, e.g. objects representing the Quality-of-Experience of VFs or the load in VMs, which are comparable between operating systems. DIL's Data

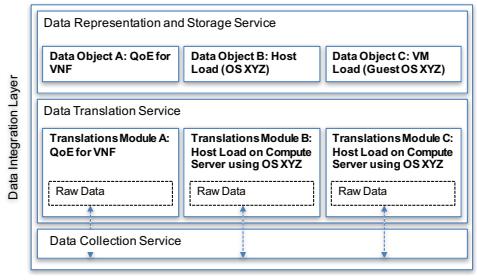

Fig. 7: Data Objects and Translation Services within the Data Integration Layer

Translation Service will comprise of a number of software modules, which translate raw load data collected from the CN components into interoperable data. The CN Control Layer, cf. Figure 6, consists of *a)* a Cloud Orchestrator (e.g. an OpenStack controller), *b)* a Network Controller (e.g. a FloodLight SDN controller), *c)* a CN Profiler, and *d)* eventually a CN (umbrella) controller, which synchronises between the other controllers. The CN profiler provides capabilities to display and correlate gathered data, such that performance bottlenecks, efficiency leaks or unused hidden resources that can be manually or automatically be identified. The output from the profiler can be fed into the other controller.

V. Conclusion and Outlook

In this contribution we discussed the capabilities adversaries of on-board Linux tools for resource monitoring in CN and NFV infrastructures. In addition, we provided a taxonomy of resources in CN infrastructures that might become bottlenecks. Furthermore, we introduced a RM architecture that enables vertical and horizontal RM in CN systems.

It turned out that the standard resource monitoring tools are able to observe utilisation on small time scales, but the trustfulness of the tools depends on the nature of the load and on the load level. We expect that additional research is necessary on the capabilities of the tools, to enhance their monitoring accuracy, and on the impact of resources such as memory or I/O. Moreover, future research should be also on identifying of improved load profiles and benchmark loads, such that RM in CN becomes more reliable.

References

[1] T. Anderson, L. Peterson, S. Shenker, and J. Turner. Overcoming the internet impasse through virtualization. *Computer*, 38(4), April 2005.

[2] T. Benson, A. Akella, A. Shaikh, and S. Sahu. Cloudnaas: a cloud networking platform for enterprise applications. In *Proc. of the 2nd ACM Symposium on Cloud Computing*, page 8. ACM, 2011.

[3] Canonical Ltd. Ubuntu. Ubuntu. Information at http://www.ubuntu.com/, 2015.

[4] N.M. Mosharaf Kabir Chowdhury and Raouf Boutaba. A survey of network virtualization. *Comput. Netw.*, 54(5), April 2010.

[5] City Network Hosting AB. Agility Through Open IT-Infrastructure. Information at https://www.citynetworkhosting.com, 2015.

[6] P. Costa, M. Migliavacca, P. Pietzuch, and A. Wolf. Naas: Network-as-a-service in the cloud. In *Proc. of the 2nd USENIX Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services*, Nov. 2012.

[7] Edsger W Dijkstra. On the role of scientific thought. In *Selected writings on computing: a personal perspective*. Springer, 1982.

[8] ETSI. Network Functions Virtualisation – An Introduction, Benefits, Enablers, Challenges & Call for Action. Information available at http://portal.etsi.org/NFV/NFV_White_Paper.pdf, 2012.

[9] W. Felterand A. Ferreira, R. Rajamony, and J. Rubio. An updated performance comparison of virtual machines and linux containers. In *Proc. 2015 IEEE International Symposium Onof the Performance Analysis of Systems and Software (ISPASS)*, 2015.

[10] V. Gite. How To Stress Test CPU and Memory (VM) On a Linux and Unix With Stress-ng. nixCraft – Linux Howto's Guide – Information available at http://www.cyberciti.biz/faq/stress-test-linux-unix-server-with-stress-ng/, Sep. 2015.

[11] S. Godard. Sysstat Utilities Home Page! Information and code available at http://sebastien.godard.pagesperso-orange.fr/index.html, 2015.

[12] A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori. kvm: the linux virtual machine monitor. In *Proceedings of the Linux Symposium*, volume 1, pages 225–230, 2007.

[13] W. LeFebvre. Top. Information and code available at http://sourceforge.net/projects/unixtop/, 2003.

[14] The CentOS Project. CentOS. Information at https://www.centos.org, 2015.

[15] R. Uhlig, G. Neiger, D. Rodgers, A. L. Santoni, F. C. M. Martins, A. V. Anderson, S. M. Bennett, A. Kagi, F. H. Leung, and L. Smith. Intel virtualization technology. *Computer*, 38(5), 2005.