Teaching Microservices in the Private Cloud by Example of the eduDScloud

Dominik Schöner, Arne Koschel, Felix Heine

Faculty IV, Department of Computer Science
University of Applied Sciences and Arts
Hannover, Germany
Email: dominik.schoener@hs-hannover.de
Email: arne.koschel@hs-hannover.de
Email: felix.heine@hs-hannover.de

***Abstract*—Cloud computing has become well established in private and public sector projects over the past few years, opening ever new opportunities for research and development, but also for education. One of these opportunities presents itself in the form of dynamically deployable, virtual lab environments, granting educational institutions increased flexibility with the allocation of their computing resources. These fully sandboxed labs provide students with their own, internal network and full access to all machines within, granting them the flexibility necessary to gather hands-on experience with building heterogeneous microservice architectures. The eduDScloud provides a private cloud infrastructure to which labs like the microservice lab outlined in this paper can be flexibly deployed at a moment's notice.**

Keywords–education; private cloud; virtual lab; microservices; SOA; eduDScloud.

I. Introduction

After cloud computing in general and private cloud computing in particular crested the 'Peak of Inflated Expectations' in Gartner's 2010 Emerging Technologies Hype Cycle [1], the technology has become widely accepted by and used not only in the industry. But due to the versatility of its basic concepts, cloud computing not only offers opportunities for commercial, but for educational and research applications as well.

At institutions focussing on education and research, it is not uncommon that computing resources for use by individual students or groups are very limited, with most resources only being available in the form of shared, pre-installed environment, often used by multiple lectures at a time. While this essentially reduces cost for hardware and personnel, it also requires very restrictive permissions to be imposed on student accounts, in order to prevent users from affecting each other, effectively limiting the possibilities for exercises and projects.

When teaching microservices meanwhile, granting students administrative access to certain systems can be essential in allowing them to gather first-hand experience on the effects of changes to system and network configuration. For scenarios like these, cloud computing can offer a way to provide labs individual, virtual machines (VMs) and networking in sandboxed environments on dynamically assignable and expandable resources, which can easily be repurposed between lectures or labs.

An example of this is the eduDScloud (*edu*cational *d*ata analytics and *s*ervice-oriented architecture *cloud*), which is used at the University of Applied Sciences and Arts Hannover in lectures and labs on data analytics and distributed systems. This paper describes the architecture of the eduDScloud using the example of a virtual lab developed for teaching service-oriented and especially microservice architectures.

The remainder of this paper is structured as follows: Section II summarises related research followed by an overview of the requirements posed by a microservice lab scenario in Section III. The system's architecture is described in Section IV, with Section V detailing the setup of the microservice lab itself and the technologies used therein. In Section VI, the findings of this paper are then summarised, before an outlook on future research is given in Section VII.

II. Related Work

The scenarios in which institutions make use of cloud computing in their educational programmes are becoming ever more diverse. Ranging from generic, collaborative Software-as-a-Service (SaaS) platforms like Google's G Suite for Education [2], to specialised platforms, e.g., for teaching high-performance computing [3] or R and Scilab [4].

Other platforms like CloudIA, proposed in [5], offer a combination of SaaS, Platform-as-a-Service (PaaS) and Infrastructure-as-a-Service in a single system, capable of running in a hybrid cloud context, scaling out to Amazon Web Services as needed. Their focus in PaaS lies within the on-demand deployment of individual VMs using a self-service portal accessible by students and lecturers. But without the concept of combining them into labs with customised, internal networking, it is impractical to use for the purpose of setting up a fully featured microservices lab.

III. Requirements

To reach the goal of providing students with a virtual hands-on lab for exploring both classic service-oriented, as well as microservice architectures a number of requirements (labelled RQ1 through RQ8 for easier reference) must be met by different parts of the overall system. While many of these are imposed by the services scenario itself, some also originate from the infrastructure surrounding the cloud itself.

Although there is no single, agreed upon, complete definition of the term microservice, many authors agree that the autonomy and technology heterogeneity of microservices is a key requirement to providing lose coupling and ease of deployment [6] [7]. Offering these two characteristics in a lab requires it to run multiple machines (RQ1) to which students have full root access (RQ2), allowing them to either install

 ISBN: 978-1-61208-606-4

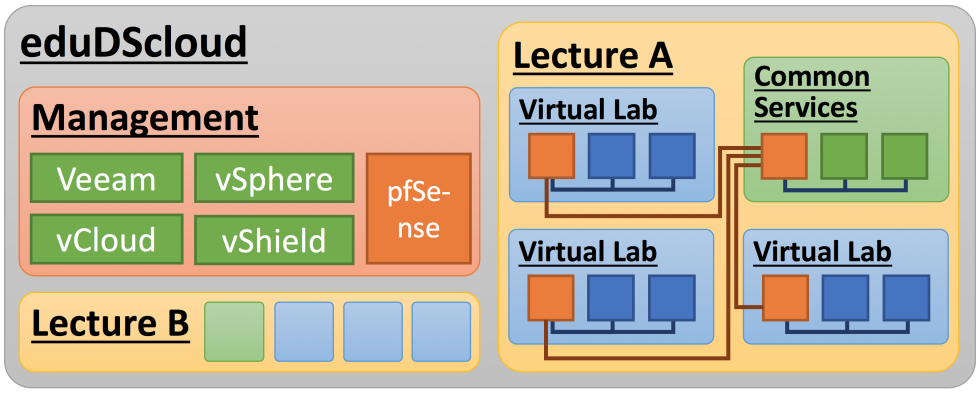

Figure 1. Schematic overview of the eduDScloud's overall architecture, detailing the connectivity between individual virtual labs (blue) and common services (green) within a lecture. The pfSense nodes (orange) can optionally also act as gateways to the internet or other internal networks.

arbitrary components or deploy containers (e.g., Docker) for individual microservices.

When providing users with extensive privileges like the aforementioned root level access, it is crucial to keep each virtual lab instance from adversely affecting others (RQ3), while still enabling full network access between its machines, as well as those of the students (RQ4). Additionally, it should be possible to create limited connectivity between individual labs for integration exercises (RQ5) and reset a lab instance in case of fatal misconfiguration (RQ6).

Since the focus of the microservices lab is on teaching service-oriented architecture (SOA) in general and microservice architectures in specific, it should also provide technologies for service discovery, monitoring and resilience (RQ7).

In order to reduce administrative efforts necessary to create and run arbitrary number of lab instances, the system should integrate with pre-existing authentication infrastructure (e.g., Lightweight Directory Access Protocol (LDAP)) (RQ8).

IV. System Architecture

The eduDScloud in its current implementation is based on VMware vCloud Director (vCD) [8] and accompanying products like ESXi (Hypervisor) and vSphere, due to pre-existing infrastructure and experience using this technology stack. As this introduces certain platform-specific limitations and requirements, the architectural concept of the eduDScloud system (see Figure 1) has been designed such as to prevent a vendor lock-in. This is achieved by avoiding dependencies on platform-specific features where possible, keeping the concept and major parts of lab setups – especially the virtual machines themselves – portable between different vendors.

A lab in vCD is reflected in the form of a vApp, which consists of one or more VMs (RQ1) and their networking – including internal networks as well as connectivity to external networks. Each lab therefore has its own internal network connecting its VMs to each other (RQ3) without any of them being directly accessible from outside the lab. The only exception to this is a pfSense-based [9] gateway VM present in all eduDScloud labs, which acts as a gateway and OpenVPN server, allowing students to join the internal network and providing access to the internet (RQ4).

Connectivity between otherwise independent vApps can be provided by addition of a *CommonServices* vApp to a lecture, which will create site-to-site VPN connections to all other labs in the lecture. While *CommonServices* can itself provide additional services, its pfSense instance also allows connected labs to offer services using network address translation (RQ5).

Both vCD and pfSense allow the use of authentication providers (RQ8) – like the LDAP infrastructure present at the University of Applied Sciences and Arts Hannover– with the eduDScloud using the permission system within vCD to manage authorization as the directory is provided as read-only resource. Permissions for individual vApps are then synced to the respective pfSense instances using a Java client which accesses the vCD API. As an alternative approach, authorization could be handled completely via LDAP – even if the directory used for authentication is read-only – by employing an intermediary identity and access management solution like e.g. Keycloak [10], which is controlled by the eduDScloud administrators.

All vApps are instantiated from templates stored in a catalogue, which can be shared between lectures as necessary. Each template represents a full snapshot of all VMs that are part of a lab, making it easy to deploy additional instances as well as reset existing ones to their original state in case of a fatal misconfiguration (RQ6). As an alternative to this potentially destructive operation, which would result in a loss of all work which has not been backed up beforehand, it is also possible to create snapshots of deployed lab instances before performing risky operations.

V. Microservices Lab

The lab created for teaching microservices is comprised of seven VMs in total, not counting the obligatory gateway VM. While the *Services-ESB* VM is an artifact of the same lab also being used in general SOA exercises, the other six VMs are either dedicated for use with a microservice architecture based on the Netflix Open Source Software stack (Netflix OSS) [11] or at least intended for dual-use to save resources.

A. Netflix OSS Stack

With regards to open source microservice stacks, Netflix OSS is mentioned by [7] for its easy accessibility on the implementation side through Spring Cloud Netflix and providing many features catering specifically to the needs of microservice architectures. For the sake of reduced complexity

 ISBN: 978-1-61208-606-4

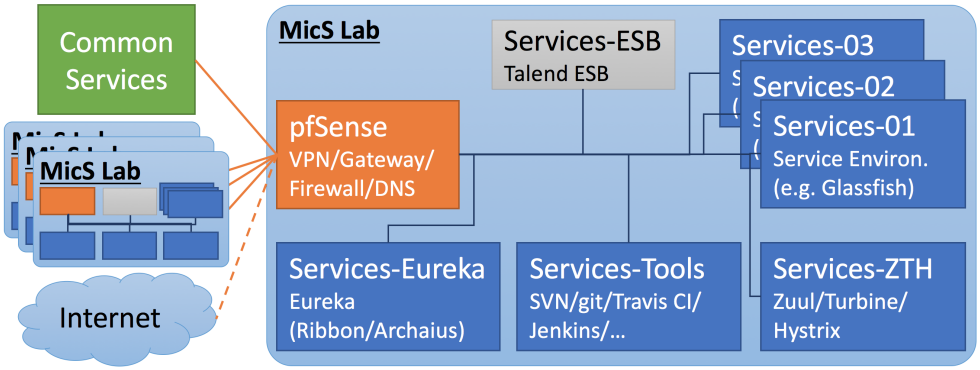

Figure 2. Diagram of the virtual microservices (MicS) lab showing all VMs and their connectivity with services and networks outside of the lab.

in an educational context, the components can be reduced to a set of four core technologies, crucial for teaching the basic characteristics of microservices (RQ7):

- *Eureka* — Provides a registry which can be used by services to locate other available services; checks availability via heartbeat.
- *Zuul* — The gateway to the entire service architecture; transparently routes requests to available service instances discovered through Eureka.
- *Turbine* — A stream aggregator used to collect metrics from all currently running service instances and make them available to Hystrix.
- *Hystrix* — Increases the architecture's resilience by offering circuit breaking capabilities [6] [12] to trigger fail-fast behaviour for services monitored via Turbine; helps prevent errors in monitored services from cascading across the entire system.

Additional components like Archaius (remote configuration) and Ribbon (load balancer) could be included for certain specific exercise tasks, but would exceed the scope of an introductory lab. Especially Ribbon also provides only very limited advantages in a lab environment, as Zuul already provides basic, round-robin load balancing.

B. VM Configuration

As can be seen in Figure 2, three of the seven VMs in the microservices lab setup are general purpose VMs (*Services-01* through *Services-03*). These VMs contain a basic installation of a Linux system along with a preconfigured application server (Glassfish) and can be increased or decreased in number as required by exercises and projects. While the application server is intended for use in exercises targeting conventional, SOAs based on the use of an enterprise service bus (ESB), these machines can also be used as fully reconfigurable and customisable hosts for individual microservices.

For this purpose, all systems offer a privileged (sudoer; root level access) and an unprivileged user account each, to both of which the students have full access (RQ2). This allows them to not only deploy services to the application server, but also install completely independent technology stacks on each host, e.g., running a Java service using PostgreSQL on one and a .NET Core service backed by a MongoDB on another. While the privileged account is needed for the necessary changes to the system, unprivileged user accounts are to be used to actually run the individual tools and services, teaching students some of the basic best practices in server administration.

One of the fully pre-configured VMs (*Services-ZTH*) in the setup is used to run a Zuul as well as a combined Turbine and Hystrix instance, whose configuration in a simplified lab environment can be considered trivial. The only changes necessary is the addition of new services to the list of monitored ones as students deploy them to their lab.

An additional, stand-alone VM (*Services-Eureka*) acts as the service registry, running a pre-configured Eureka instance, which is already hooked up with Zuul. In addition, this machine also offers basic preparations for deploying Archaius and Ribbon, should specific projects or exercises require them.

To facilitate source code versioning as well as automated build and deployment processes, the *Services-Tools* VM – in its most basic configuration – acts as a Subversion and git host. Depending on the deployment toolchain [13] chosen for the lab, it can quickly be set up to run a choice of build and test automation tools like Jenkins or Travis CI. When the lab is run with a focus on conventional SOA meanwhile, this host also acts as a database server, allowing *Services-ZTH* and *Services-Eureka* to be shut down in order to save resources.

The last VM in the lab's line-up is *Services-ESB*, which – in its current configuration – runs an instance of the Talend ESB, capable of deploying OSGi containers via Apache Karaf. This combination of ESB-based SOA and technologies focused on microservices within a single lab allows lecturers to not only teach these concepts independently from each other, but also in tandem, e.g., in the form of integration challenges.

C. First Experiences with the Lab

The current setup running at the University of Applied Sciences and Arts Hannover includes total of 28 cores (48 threads), having access to 512 GB of RAM and 40 TB of storage for the actual cloud itself. The management server, which is running infrastructure services like vCloud, vSphere, vShield (networking), pfSense (DNS and DHCP) and a backup solution called Veeam [14], is equipped with 4 cores (8 threads), 32 GB of RAM and 4 TB HDD storage. For backups, additional space is supplied by a QNAP storage array providing an additional 8 TB of disk space, to which snapshots of the management VMs and lab templates are written.

 ISBN: 978-1-61208-606-4

Due to the number of VMs contained within a single instance of the virtual microservices lab, the number of cores in the initial hardware setup has turned out to quickly become an issue. With between three to four students per group and each VM (except for the pfSense instance) using two virtual cores (vCores) each, a lecture with 25 students needs about one hundred vCores in total. This makes it impractical to run multiple lectures in parallel on the current hardware, which can be partially alleviated by temporarily suspending labs of other lectures for the duration of certain exercise sessions.

Some issues that occurred during the exercises themselves were, that OpenVPN – used to connect with the virtual lab environments internal network – requires administrative permissions on the client system. Since providing students with root level access to shared pool systems is infeasible, a custom script had to be implemented in order to allow unprivileged users to run OpenVPN using custom configurations and temporarily modify the clients' DNS configuration.

VI. Conclusion

The eduDScloud provides a solid, scalable private cloud solution for hosting virtual labs used in teaching microservices and other topics from the field of computer sciences. It allows students to gain hands-on experience by providing them with root access to virtual machines and networks within a sandboxed environment. At the same time, it grants lecturers and tutors the ability to devise more complex exercise tasks with a minimum of administrative work as new labs can easily be instantiated from pre-built templates.

A specialised lab setup has been presented as an example for a virtual lab running on the eduDScloud, used to teach concepts of SOA in general and microservices in specific. This lab provides students with a preconfigured stack for ESB-based service development as well as running the Netflix OSS stack to showcase microservice-specific technologies and patterns. The presence of both stacks in a single lab, along with the interconnectivity of deployed lab instances also opens up opportunities for integration exercises between student groups, which represent a common real-world use case for service-oriented architectures.

VII. Future Work

While the eduDScloud is already capable of scaling out across multiple servers, its current implementation limits it to private cloud capabilities. This can become especially challenging when, e.g., due to scheduling issues, a high number of lab instances has to run in parallel, potentially exceeding the computing resources of the available hardware. To alleviate these issues, upcoming projects include migrating the eduDScloud towards a hybrid cloud concept, allowing it to scale out using public cloud resources on demand, while maintaining privacy protection for sensitive applications.

Part of this migration is the evaluation of open-source virtualisation platforms and frameworks such as Proxmox [15] and especially OpenStack [16], as the latter would allow seamless integration with various public cloud providers. Switching to one of these solutions would also allow for the entire stack of the eduDScloud to be running solely on open-source software, as VMware components are currently the only proprietary, closed-source part of the system.

In addition to this, further research topics include the development of new lab concepts for other fields of computer science, increasing the platform's resilience and improving resource scheduling. The latter is currently handled manually and is planned to be automated by synchronising it with a live schedule for labs and lectures, and integrating it with a reservation platform to handle projects and other activities.

Acknowledgement

The authors would like to thank the students which were part of the combined bachelor and master project during the winter and summer semester 2016/2017 for their contribution to the implementation of the eduDScloud.

References

[1] J. Fenn. 2010 Emerging Technologies Hype Cycle is Here - Mastering The Hype Cycle. [retrieved: 2018-01-05]. [Online]. Available: https://blogs.gartner.com/hypecyclebook/2010/09/07/2010-emerging-technologies-hype-cycle-is-here/ (2010)

[2] Google. Higher Ed G Suite — Google for Education. [retrieved: 2018-01-05]. [Online]. Available: https://edu.google.com/higher-ed-solutions/g-suite/

[3] S. S. Foley, D. Koepke, J. Ragatz, C. Brehm, J. Regina, and J. Hursey, "Onramp: A web-portal for teaching parallel and distributed computing," Journal of Parallel and Distributed Computing, vol. 105, 2017, pp. 138–149.

[4] K. Chine, "Learning math and statistics on the cloud, towards an ec2-based google docs-like portal for teaching / learning collaboratively with r and scilab," in 2010 10th IEEE International Conference on Advanced Learning Technologies. IEEE, 2010, pp. 752–753.

[5] F. Doelitzscher, A. Sulistio, C. Reich, H. Kuijs, and D. Wolf, "Private cloud for collaboration and e-learning services: From iaas to saas," Computing, vol. 91, no. 1, 2011, pp. 23–42.

[6] S. Newman, Building Microservices, 1st ed. Sebastopol: O'Reilly & Associates, 2015.

[7] E. Wolff, Microservices: Grundlagen flexibler Softwarearchitekturen, 1st ed. Heidelberg: dpunkt.verlag, 2016.

[8] VMware. Deliver Virtual Data Centers — vCloud Director — VMware. [retrieved: 2018-01-05]. [Online]. Available: https://www.vmware.com/products/vcloud-director.html

[9] Rubicon Communications. pfSense - World's Most Trusted Open Source Firewall. [retrieved: 2018-01-05]. [Online]. Available: https://www.pfsense.org

[10] I. Red Hat. Keycloak. [retrieved: 2018-01-05]. [Online]. Available: http://www.keycloak.org

[11] Netflix. Netflix Open Source. [retrieved: 2018-01-05]. [Online]. Available: https://netflix.github.io

[12] S. J. Fowler, Production-ready microservices: Building standardized systems across an engineering organization, first edition ed. Beijing and Boston and Farnham and Sebastopol and Tokyo: O'Reilly, 2016.

[13] T. Hunter II, Advanced Microservices: A Hands-on Approach to Microservice Infrastructure and Tooling, 1st ed. New York: APRESS, 2017.

[14] Veeam Software. Veeam Availability for the Always-On Enterprise. [retrieved: 2018-01-05]. [Online]. Available: https://www.veeam.com

[15] Proxmox Server Solutions GmbH. Proxmox - powerful open-source server solutions. [retrieved: 2018-01-05]. [Online]. Available: https://www.proxmox.com/en/

[16] OpenStack Foundation. Openstack Open Source Cloud Computing Software. [retrieved: 2018-01-05]. [Online]. Available: https://www.openstack.org

 ISBN: 978-1-61208-606-4