T4.1 Deployment Services -- CICD

https://portal.azure.com/#@austriandatalab.onmicrosoft.com/resource/subscriptions/d3178f52-bf32-4360-a534-5f4faa991f62/resourcegroups/E020-04-Terraform-Backend/providers/Microsoft.Compute/virtualMachines/terraform/overview

What is the core value being generated?	Team	Status
Time savings through Automation Transparency through Reproducibility Setup (one time): First integration of all tools in the Zoo	**Project owner / Deputy owner:** Constanze Rödig **Team members:** Constanze, Elias	ACTIVE

Problem space

Why are we doing this?	**Problem statement (these are assumptions, please tell Constanze if not correct)** Container orchestrator (e.g. k8s) needs to be deployed routinely in order to run services on top. Services that run inside the orchestrator need to be deployed routinely. Quality of a full-stack deployment (e.g. stability and security, applicability of a license) is not transparent to a new joiner Deployments are not yet fully automated, e.g. command line execution is used, servers are being sshed into etc Knowledge sharing is dependent on an expert user and thus does not scale **Impact of this problem** Trouble shooting someone else's deployment is hard as knowledge is not readily available Changing or repeating a deployment has manual effort and takes time ("monkey time") Observability of how standards are implemented is not given. **Who is the customer/ target audience** MVP target audience is Team T2.1 (Jupyter)
How do we judge success?	Deployments become commodity and are 100% automated Non functional standards are being tested and are observable. Knowledge can be acquired through self-service (see target audience = technical people)

Minimal viable product/service ("MVP")

What needs to be true in order for a prototype to be ready for release?	We can release our MVP to Team T2.1, as soon as we have **The following MUST-HAVE features** 1. P1CICD IDE hosted accessible to users (**PaaS**) 2. P2Automation tasks running on agents that are integrated with the CICD IDE (**PaaS**) 3. P3Pipelines that contain the recipies, tasks and steps of the automation tasks to serve as **BluePrint** 4. P4DAST and SAST **services** (at least one each) that demo the integration of security services 5. P5At least one dummy hello world service that is deployed using 100% automation to serve as **BluePrint** 6. P6At least on runtime (likely k8s) deployed to CloudProvider via 100% automation, aka IaC (Infrastructure as Code) to serve as **BluePrint** **The following non-functionals are MUST-HAVE:** 1. P7Users are authenticated, no unauthenticated access (**PaaS**) 2. P8The base images of any runtime (both IaaS as well as Containers) are patched /updated at least once a week, failure thereof will result in an Alert (**BluePrint**) 3. P9Usage of (static) vulnerabilities higher than 7.5 are flagged as red, but are non-blocking, a dashboard exists (**BluePrint)** 4. P10Policy for branch naming exists (**BluePrint)** 5. P11Policy for peer review of Pull Requests exist (**BluePrint**) 6. P12An integration with Wiki tool exists (**PaaS**) 7. P13An integration with Task tool exists (**PaaS**) **Not in scope are:** 1. We will only use one cloud provider at this stage, no generalization to other clouds 2. Seamless integration of the CICD IDE to kubernetes has strong network/dns/firewall dependencies, thus only Networks will be used, that are in our control. 3. The deployment of the Cloud Provider itself is not in scope of this MVP, a separate one will be created **if** there is interest to use OpenStack at other institutes 4. Single Sign On (SSO) , Integrated Access Management 5. any SLAs
What crucial factors are we missing?	Strong dependency on the OpenStack deployment being available for agents Constanze doesnt necessarily understand the DNS zones, help from TUWien Network team or Elias is required Firewall settings need to be revisited, help from TUWien Network team and/or Elias is required AuthN is assumed to be solved "somehow" (ADRs will be needed) Task Tool is not yet available Vault for Secrets is needed, not yet available (ADRs needed)

Continued Feedback

What is the key question we would ask to understand if we are on the right track?	How much time did you save using automated CI/CD compared to running ansible scripts/ cli commands before? Do you find the solution increases transparency along the various dimensions (howto, performance, security, IP)?
Who are the alpha testers that we can use for validating our assumptions?	Elias Wimmer

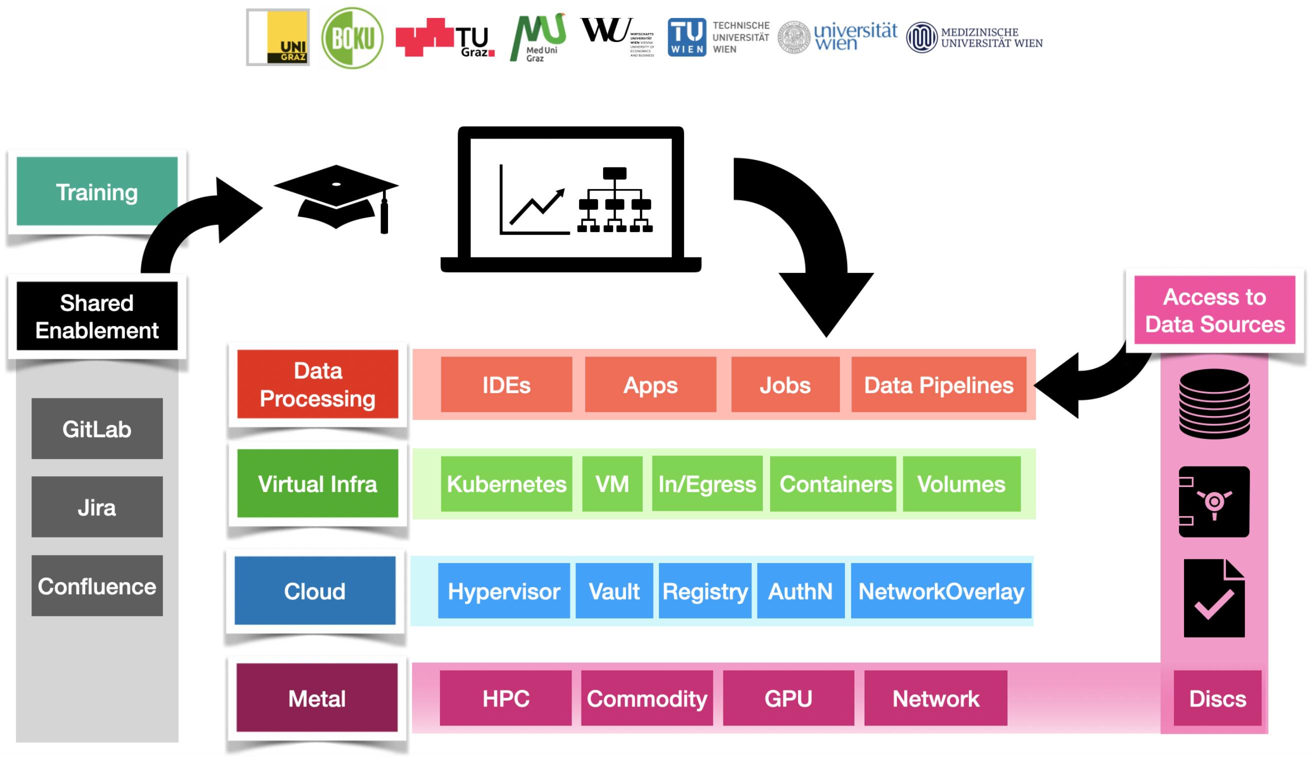
Shipment 2020-12

Original Goal	Deliverable	Issues encountered	Links to deliverable	Demo scheduled for

Summary	A sample IaC deployment Incl agents, IAM and sample apps was created. Main purpose: 1. Have a working Azure setup which all of us can use, IF it becomes relevant 2. Have the ability to demo how to use PubCloud securely			
P1 CICD IDE hosted accessible to users (**PaaS**)	gitlab.tuwien.ac.at has ADLS project, most ADLS users are added	there seem to be some bugs with the initial CI setups	https://gitlab.tuwien.ac.at/ADLS/samples/demo_cicd ADR: ADR-0002 IDE for CI/CD	implicit Dec 2020 D1 - Monthly Demo
P2Automation tasks running on agents that are integrated with the CICD IDE (**PaaS**)	The Runners exist (on demand) in Azure.	there seem to be some bugs with the initial CI setups	Concept is here: Access Concept - Azure lots of pipelines are running e.g. here: https://gitlab.tuwien.ac.at/ADLS/samples/demo_cicd/-/pipelines	D1 - Monthly Demo
P3Pipelines that contain the recipies, tasks and steps of the automation tasks to serve as **BluePrint**	A ReadMe on how to use it for your own work will be ready by Demo	I wanted a real password free, fully user friendly build method. Thus, I needed to move my Azure setup a total of 4 times, because the early (TUW) setups were too restrictive. Now, we have our own ADLS tenant, which we own.	this is a working example, but I ll update it with a better one until Demo: https://gitlab.tuwien.ac.at/ADLS/samples/demo_cicd/-/blob/master/app/cicd/devops/deploy/keycloak/.gitlab-ci.yml	
P4DAST and SAST **services** (at least one each) that demo the integration of security services	SAST: starboard at Deploy Time DAST: ran out of time , had to descope it	DAST:not enough time (see above) and since DAST is the most sophisticated of all features, I descoped it in favour of the fundamentally important IAM setup	Still need to write a nice ReadMe and create a custom alert based on the output of https://aquasecurity.github.io/starboard/operator/ (which is auto installed on our ADLS-K8s)	D1 - Monthly Demo
P5At least one dummy hello world service that is deployed using 100% automation to serve as **BlueP rint**	I m actually deploying Keycloak as a dummy service		https://gitlab.tuwien.ac.at/ADLS/samples/demo_cicd/-/blob/master/app/cicd/devops/deploy/keycloak/.gitlab-ci.yml	

P6At least on runtime (likely k8s) deployed to CloudProvider via 100% automation, aka IaC (Infrastructure as Code) to serve as **BluePrint**	Loosely coupled AZ deployemnt includes: - manual steps if you start from nothing - full automation via terraform to create 31 infrastructure components: 1. networks, dns record, firewall, peering 2. k8s cluster with 2 pools 3. keyvaults and access policies 4. roles and role bindings 5. identities and identity bindings	not tested since Oliver left	ADR: ADR-0007 Deployment Automation Paradigm IaC IaC _1 pipeline including k8s roles: https://gitlab.tuwien.ac.at/ADLS/samples/demo_cicd/-/jobs/2096	D1 - Monthly Demo
P7Users are authenticated, no unauthenticated access (**PaaS**)	**Infrastructure incl Vaults:** MFA using Azure AD, Onboarding for Guest of all unis possible **Applications:** Keycloak and Grafana have (preset) passwords	OIDC for the apps is very possible, need to decide on the AuthN federation There are many options to do it, but a service catalogue and access pattern will help determine the chosen solution.	Concept for users to access Infrastructure components: IAM Concept Azure Implementation of User Roles (self-tested): https://gitlab.tuwien.ac.at/ADLS/samples/demo_cicd/-/jobs/2145 Implementation of Pod Identity (untested): https://gitlab.tuwien.ac.at/ADLS/samples/demo_cicd/-/jobs/2146	can be demoed if interest exists, its pretty technical
P8The base images of any runtime (both IaaS as well as Containers) are patched /updated at least once a week, failure thereof will result in an Alert (**BluePrint**)	Terraform Agent VM: updates itself everyday, but no alerts Kubernetes: solution identified, not deployed	ran out of time clusters recreate every day, so risk is not considered large.	Terraform Agent VM: (you will likely not be able to login, but the machine exists and updates itself) https://portal.azure.com/#@austriandatalab.onmicrosoft.com/resource/subscriptions/d3178f52-bf32-4360-a534-5f4faa991f62/resourcegroups/E020-04-Terraform-Backend/providers/Microsoft.Compute/virtualMachines/terraform/overview NOT DONE: Kubernetes: *helm install kured kured/kured*	
P9Usage of (static) vulnerabilities higher than 7.5 are flagged as red, but are non-blocking, a dashboard exists (**BluePrint)**	Starboard flags them as red, is non blocking and octant can be used as client side washboarding tool		Until Demo: insert pictures into ReadMe	D1 - Monthly Demo will quickly show the combo Octant /Starboard
P10 Policy for branch naming exists (**BluePrint)**	Did exist but was removed due to bugs	Thomas Weber reported that the branching policy interfered and made his entire repo unusable, thus removed until bug understood. not considered critically necessary		

P11Policy for peer review of Pull Requests exist (**BluePrint**)	It exists and works. Minimal reviewer=1	pretty basic, didn't have any co-workers for review, so couldn't test more fancy models		
P12An integration with Wiki tool exists (**PaaS**)	Gitlab can connect to this Confluence		gitlab External Wiki	
P13An integration with Task tool exists (**PaaS**)	Gitlab can see JIRA, but the PROD Jira is coming mid-January	external dependency on JIRA delivery		


UNI GRAZ
BOKU
TU Graz
Med Uni Graz
WU
TU WIEN TECHNISCHE UNIVERSITÄT WIEN
universität wien
MEDIZINISCHE UNIVERSITÄT WIEN
Training
Shared Enablement
GitLab
Jira
Confluence
Data Processing
IDEs
Apps
Jobs
Data Pipelines
Virtual Infra
Kubernetes
VM
In/Egress
Containers
Volumes
Cloud
Hypervisor
Vault
Registry
AuthN
NetworkOverlay
Metal
HPC
Commodity
GPU
Network
Access to Data Sources
Discs