Inhaltsangabe

Ziel: Cloud Computing System mit integrierten Selbstheilungskonzepten

- OpenStack
- mögliche Fehlerfälle in einer Cloud Computing Umgebung analysieren
- Entwicklung eines Prozess, der Daten aus dem System sammelt, analysisert und im Fehlerfall reagiert

Produkt: Cloud Platform (OpenStack), auf dem Nutzer Applikationen Hosten können (Bereitstellung einer Cloud-Lösung für Nutzer)

Empfehlung: kolla-ansible

- auf den 4 Servern soll OpenStack laufen
- Nutzer kann VMs starten
- Nutzer kann virtuelle Netzwerke nutzen
- Nutzer kann VMs von „außen“ erreichen (erstmal vom OpenStack Controller Node)

NOCH 5 WOCHEN

1. Gruppenbildung & Organisation
2. Vortrag über ein Thema
3. Gruppenorganisation & Zugriffsbereitstellung
4. Planung
5. Ergebnispräsentation
6. Planung
7. Meilensteinpräsentation
8. Planung
9. Ergebnispräsentation
10. Planung
11. Abschlusspräsentation 18.02.

Anfang
OpenStack Setup – Eine kleine Hilfestellung (ISIS-Beitrag)

hier kann man es nachlesen:
https://docs.openstack.org/project-deploy-guide/kolla-ansible/ussuri/quickstart.html

Allgemein:
- Ansible-Playbooks und kolla-ansible Befehler lokal auf eigenem Rechner laufen lassen
- im Inventory auf welche Knoten (die wally-Server) die Tasks laufen sollen definieren
  - den User, den Ansible nutzen soll, wenn es sich per ssh mit den Nodes verbindet, in der ansible.cfg oder im Inventory definieren (falls notwendig, den Pfad zum private key ebenfalls)
  - Verbindung möglich? –> ping-Befehl benutzen: https://docs.ansible.com/ansible/2.3/ping_module.html

Pre-bootstrap:
- 2. Interface einrichten & testen, ob unsere Knoten überhaupt erreichbar sind ✅
- 1) a. netplan-Modul (wenn ansible) nutzen ODER
     b. netplan config per jinja-Template ausfüllen
  2) auf Knoten kopieren
  3) netplan apply ausführen
- folgende Pakete müssen auf Knoten installiert sein (vor bootstrap von kolla-ansible):
  - python3
  - python3-pip
  - python3-dev
  - libffi-dev
  - gcc
  - libssl-dev
- Sicherstellung (mit pip), dass docker installiert und docker-py **nicht installiert** ist

Lokal:
Empfehlung: venv, da drauf muss ansible und kolla-ansible installiert sein

Konfiguration:
Nötige Dateien (für kolla-ansible):
- globals.yml
- passwords.yml –> **nicht in unsere Repo pushen**
- multinode

Konfiguration von **globals.yml**:

1) Innerhalb der # Kolla options:
kolla_base_distro: "ubuntu"
openstack_release: "train" (Empfehlung)
kolla_internal_vip_address: "130.149.249.[control_ip]"

2) Innerhalb der # Neutron - Networking Options:
network_interface: "enp2s0"
cluster_interface: "<Name eures 2. Netzwerkinerfaces>"
neutron_external_interface: "<Name eures 2. Netzwerkinerfaces>"

3) Innerhalb von # OpenStack options:
enable_haproxy: "no"

4) In multinode:
(Empfehlung (wie Alexander das macht))
[all:vars]
ansible_python_interpreter=/usr/bin/python3
ansible_user=alexander.acker –> **Anpassen**
ansible_become=True
ansible_private_key_file={{lookup('env','HOME')}}/.ssh/id_rsa

Beispiel:
[control]
wally135.cit.tu-berlin.de –> **unser Control Node**

[network:children]
control

[compute]
wally141.cit.tu-berlin.de –> **unser Compute Node 1**
wally139.cit.tu-berlin.de –> **unser Compute Node 2**

- network soll auf control laufen

- der Rest soll auskommentiert bleiben

Bugfix:

Nach dem Ausführen von bootstrap wird in den /etc/hosts-Dateien der jeweiligen 130er IP zwei Hostnamen zugewiesen.

Das sollte dazu führen, dass die prechecks nicht klappen.

Der Eintrag, der nicht von kolla-ansible vorgenommen wurde, muss rausgenommen werden.

Ein Fix kann z.B. mit dem ansible-Modul *lineinfile* und einem entsprechenden *regex* geschrieben werden.

Ende der Eine kleine Hilfestellung (ISIS-Beitrag)

Nodes:
auf wally135.cit.tu-berlin.de Controller Node (kolla-ansible)
auf wally139.cit.tu-berlin.de (OpenStack-ansible)

für Skript-Ausführung beiseite gelegt:
einzelne Skripte testen:
wally141.cit.tu-berlin.de Compute Node 1
ein Skript anwenden auf:
wally142.cit.tu-berlin.de Compute Node 2

Erledigt:
1. VPN-Konfigurationsdatei erhalten

- mit VPN-Client können wir jetzt mit dem Uni-Subnetz verbinden und auf die Server zugreifen

z.B. mit openvpn CLI:
'sudo openvpn –config /Pfad/zur/VPNKonfig.ovpn'

2. Unseren Gruppennamen festgelegt: OurSky

- Repository auf GitHub erstellt
- Rollen im Team:

Mitglieder und Rollen

- Jonathan (Scrum master & developer)
- Oliver (developer)
- Nadia (developer)
- Zead (Product owner & developer)

ssh key-Paar erstellt und an Alexander Public Key gesendet

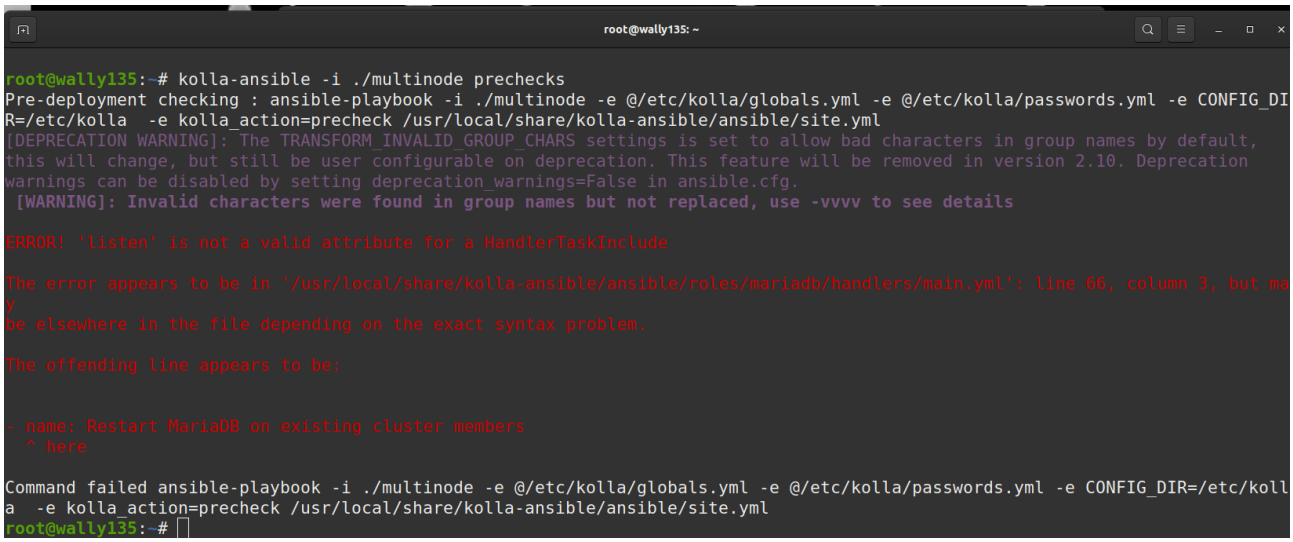
OpenStack-Rollen: Admin & Nutzer

3. Vorbereitung Control Node ✅
4. Vorbereitung Compute Node ✅
5. Etablierung von Docker als Untersystem für OpenStack ✅
6. Aktivierung & Konfiguration physischer Netzwerke ✅
7. Bootstrapping ✅
8. Prechecks ✅

In progress:
- OpenStack Deployment ❌
- kolla_toolbox container is not running

Probleme:
1.) kolla_toolbox

2.) Prechecks funktionieren nicht



Grund:

- Installierung von ansible & kolla-ansible in den Servern
- Ausführung aller Befehle in den Servern

Lösung:

- lokale Installierung
- Ausführung aller Befehle lokal

- Fehler bei Prechecks:

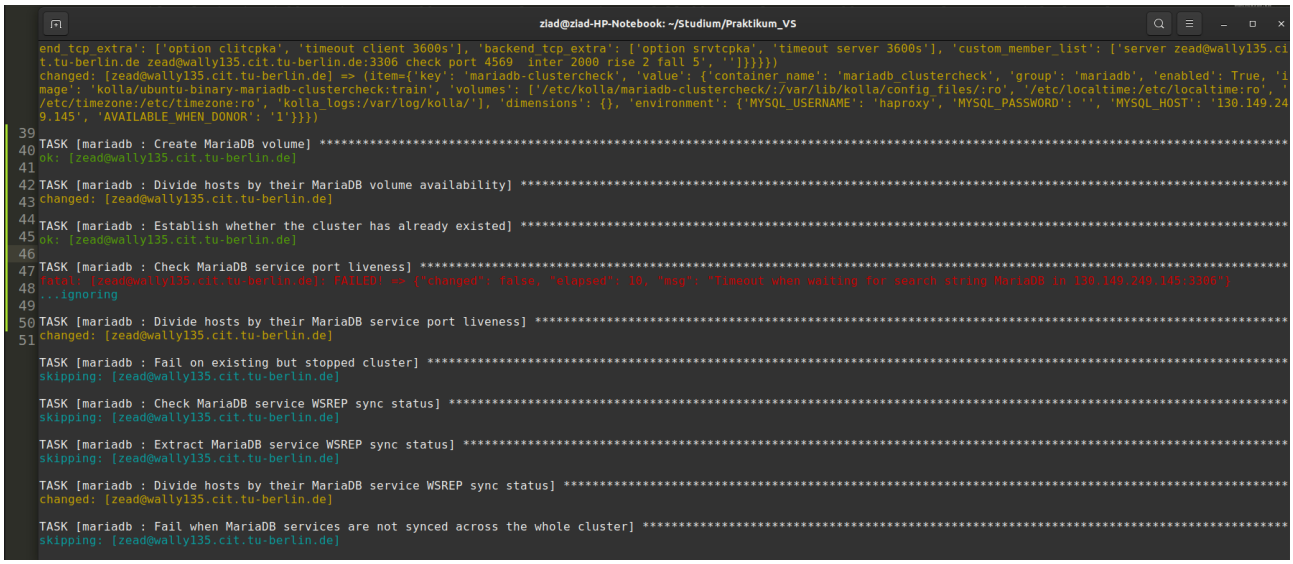
```
############################################################################
TASK [service-precheck : neutron | Validate inventory groups]
****************************************************************************
****************
fatal: [zead@wally142.cit.tu-berlin.de]: FAILED! => {"msg":
"{{ neutron_services }}: {'neutron-server': {'container_name':
'neutron_server', 'image': '{{ neutron_server_image_full }}',
'enabled': True, 'group': 'neutron-server', 'host_in_groups': \"{{
inventory_hostname in groups['neutron-server'] }}\", 'volumes':
'{{ neutron_server_default_volumes +
neutron_server_extra_volumes }}', 'dimensions':
'{{ neutron_server_dimensions }}', 'healthcheck':
'{{ neutron_server_healthcheck }}', 'haproxy': {'neutron_server':
{'enabled': '{{ enable_neutron | bool and not
neutron_enable_tls_backend | bool}}', 'mode': 'http', 'external':
False, 'port': '{{ neutron_server_port }}', 'listen_port':
'{{ neutron_server_listen_port }}'}, 'neutron_server_external':
{'enabled': '{{ enable_neutron | bool and not
neutron_enable_tls_backend | bool}}', 'mode': 'http', 'external':
True, 'port': '{{ neutron_server_port }}', 'listen_port':
'{{ neutron_server_listen_port }}'}}, 'neutron-openvswitch-
agent': {'container_name': 'neutron_openvswitch_agent', 'image':
'{{ neutron_openvswitch_agent_image_full }}',
```

```
'enabled': \"{{ neutron_plugin_agent == 'openvswitch' }}\",
'privileged': True, 'environment': {'KOLLA_LEGACY_IPTABLES':
'{{ neutron_legacy_iptables | bool | lower }}'}, 'host_in_groups':
\"{{ ( inventory_hostname in groups['compute'] or
(enable_manila_backend_generic | bool and inventory_hostname in
groups['manila-share']) or inventory_hostname in groups['neutron-
dhcp-agent'] or inventory_hostname in groups['neutron-l3-agent']
or inventory_hostname in groups['neutron-metadata-agent'] and not
enable_nova_fake | bool ) or ( inventory_hostname in
groups['neutron-dhcp-agent'] or inventory_hostname in
groups['neutron-l3-agent'] or inventory_hostname in
groups['neutron-metadata-agent'] and enable_nova_fake |
bool ) }}\", 'volumes':
'{{ neutron_openvswitch_agent_default_volumes +
neutron_openvswitch_agent_extra_volumes }}', 'dimensions':
'{{ neutron_openvswitch_agent_dimensions }}', 'healthcheck':
'{{ neutron_openvswitch_agent_healthcheck }}'}, 'neutron-
linuxbridge-agent': {'container_name':
'neutron_linuxbridge_agent', 'image':
'{{  neutron_linuxbridge_agent_image_full }}', 'privileged': True,
'enabled': \"{{ neutron_plugin_agent == 'linuxbridge' }}\",
'environment': {'KOLLA_LEGACY_IPTABLES':
'{{ neutron_legacy_iptables | bool | lower }}'}, 'host_in_groups':
\"{{ inventory_hostname in groups['compute'] or (enable_manila |
bool and inventory_hostname in groups['manila-share']) or
inventory_hostname in groups['neutron-dhcp-agent'] or
inventory_hostname in groups['neutron-l3-agent'] or
inventory_hostname in groups['neutron-metadata-agent'] }}\",
'volumes': '{{ neutron_linuxbridge_agent_default_volumes +
neutron_linuxbridge_agent_extra_volumes }}', 'dimensions':
'{{ neutron_linuxbridge_agent_dimensions }}', 'healthcheck':
'{{ neutron_linuxbridge_agent_healthcheck }}'}, 'neutron-dhcp-
agent': {'container_name': 'neutron_dhcp_agent', 'image':
'{{ neutron_dhcp_agent_image_full }}', 'privileged': True,
'enabled': \"{{ neutron_plugin_agent not in ['ovn', 'vmware_nsxv',
'vmware_nsxv3'] }}\", 'group': 'neutron-dhcp-agent',
'host_in_groups': \"{{ inventory_hostname in groups['neutron-dhcp-
agent'] }}\", 'volumes': '{{ neutron_dhcp_agent_default_volumes +
neutron_dhcp_agent_extra_volumes }}', 'dimensions':
'{{ neutron_dhcp_agent_dimensions }}', 'healthcheck':
'{{ neutron_dhcp_agent_healthcheck }}'}, 'neutron-l3-agent':
{'container_name': 'neutron_l3_agent', 'image':
'{{ neutron_l3_agent_image_full }}', 'privileged': True,
'enabled': \"{{ neutron_plugin_agent not in ['ovn', 'vmware_nsxv',
'vmware_nsxv3', 'vmware_dvs'] }}\", 'environment':
{'KOLLA_LEGACY_IPTABLES': '{{ neutron_legacy_iptables | bool |
lower }}'}, 'host_in_groups': \"{{ inventory_hostname in
groups['neutron-l3-agent'] or (inventory_hostname in
groups['compute'] and enable_neutron_dvr | bool) }}\", 'volumes':
'{{ neutron_l3_agent_default_volumes +
neutron_l3_agent_extra_volumes }}', 'dimensions':
```

```
'{{ neutron_l3_agent_dimensions }}', 'healthcheck':
'{{ neutron_l3_agent_healthcheck }}'}, 'neutron-sriov-agent':
{'container_name': 'neutron_sriov_agent', 'image':
'{{ neutron_sriov_agent_image_full }}', 'privileged': True,
'enabled': \"{{ enable_neutron_sriov | bool and
neutron_plugin_agent not in ['vmware_nsxv', 'vmware_nsxv3' ] }}\",
'host_in_groups': \"{{ inventory_hostname in
groups['compute'] }}\", 'volumes':
'{{ neutron_sriov_agent_default_volumes +
neutron_sriov_agent_extra_volumes }}', 'dimensions':
'{{ neutron_sriov_agent_dimensions }}', 'healthcheck':
'{{ neutron_sriov_agent_healthcheck }}'}, 'neutron-mlnx-agent':
{'container_name': 'neutron_mlnx_agent', 'image':
'{{ neutron_mlnx_agent_image_full }}',
'enabled': \"{{ enable_neutron_mlnx | bool and
neutron_plugin_agent not in ['vmware_nsxv', 'vmware_nsxv3' ] }}\",
'host_in_groups': \"{{ inventory_hostname in
groups['compute'] }}\", 'volumes':
'{{ neutron_mlnx_agent_default_volumes +
neutron_mlnx_agent_extra_volumes }}', 'dimensions':
'{{ neutron_mlnx_agent_dimensions }}'}, 'neutron-eswitchd':
{'container_name': 'neutron_eswitchd', 'image':
'{{ neutron_eswitchd_image_full }}', 'privileged': True,
'enabled': \"{{ enable_neutron_mlnx | bool and
neutron_plugin_agent not in ['vmware_nsxv', 'vmware_nsxv3' ] }}\",
'host_in_groups': \"{{ inventory_hostname in
groups['compute'] }}\", 'volumes':
'{{ neutron_eswitchd_default_volumes +
neutron_eswitchd_extra_volumes }}', 'dimensions':
'{{ neutron_eswitchd_dimensions }}'}, 'neutron-metadata-agent':
{'container_name': 'neutron_metadata_agent', 'image':
'{{ neutron_metadata_agent_image_full }}', 'privileged': True,
'enabled': \"{{ neutron_plugin_agent not in [ 'ovn',
'vmware_nsxv', 'vmware_nsxv3' ] }}\",
'host_in_groups': \"{{ inventory_hostname in groups['neutron-
metadata-agent'] or (inventory_hostname in groups['compute'] and
neutron_plugin_agent == 'openvswitch' and enable_neutron_dvr |
bool) }}\", 'volumes': '{{ neutron_metadata_agent_default_volumes
+ neutron_metadata_agent_extra_volumes }}', 'dimensions':
'{{ neutron_metadata_agent_dimensions }}', 'healthcheck':
'{{ neutron_metadata_agent_healthcheck }}'}, 'neutron-ovn-
metadata-agent': {'container_name': 'neutron_ovn_metadata_agent',
'image': '{{ neutron_ovn_metadata_agent_image_full }}',
'privileged': True, 'enabled': \"{{ neutron_plugin_agent ==
'ovn' }}\", 'host_in_groups': \"{{ inventory_hostname in
groups['neutron-ovn-metadata-agent'] }}\", 'volumes':
'{{ neutron_ovn_metadata_agent_default_volumes +
neutron_ovn_metadata_agent_extra_volumes }}', 'dimensions':
'{{ neutron_ovn_metadata_agent_dimensions }}', 'healthcheck':
'{{ neutron_ovn_metadata_agent_healthcheck }}'}, 'neutron-bgp-
dragent': {'container_name': 'neutron_bgp_dragent', 'image':
```

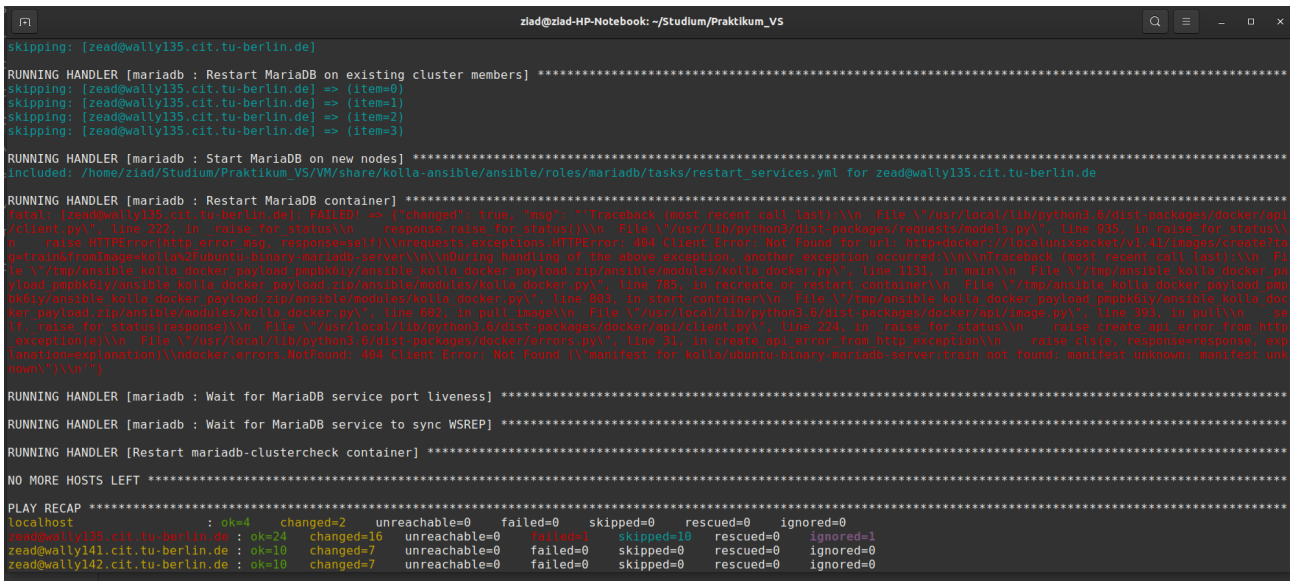
```
'{{ neutron_bgp_dragent_image_full }}', 'privileged': True,
'enabled': \"{{ enable_neutron_bgp_dragent | bool and
neutron_plugin_agent not in ['ovn', 'vmware_nsxv', 'vmware_nsxv3',
'vmware_dvs'] }}\", 'group': 'neutron-bgp-dragent',
'host_in_groups': \"{{ inventory_hostname in groups['neutron-bgp-
dragent'] }}\", 'volumes': '{{ neutron_bgp_dragent_default_volumes
+ neutron_bgp_dragent_extra_volumes }}', 'dimensions':
'{{ neutron_bgp_dragent_dimensions }}'}, 'neutron-infoblox-ipam-
agent': {'container_name': 'neutron_infoblox_ipam_agent', 'image':
'{{ neutron_infoblox_ipam_agent_image_full }}', 'privileged':
True, 'enabled': '{{ enable_neutron_infoblox_ipam_agent |
bool }}', 'group': 'neutron-infoblox-ipam-agent',
'host_in_groups': \"{{ inventory_hostname in groups['neutron-
infoblox-ipam-agent'] }}\", 'volumes':
'{{ neutron_infoblox_ipam_agent_default_volumes +
neutron_infoblox_ipam_agent_extra_volumes }}', 'dimensions':
'{{ neutron_infoblox_ipam_agent_dimensions }}'}, 'neutron-
metering-agent': {'container_name': 'neutron_metering_agent',
'image': '{{ neutron_metering_agent_image_full }}', 'privileged':
True, 'enabled': '{{ enable_neutron_metering | bool }}', 'group':
'neutron-metering-agent',
'host_in_groups': \"{{ inventory_hostname in groups['neutron-
metering-agent'] }}\", 'volumes':
'{{ neutron_metering_agent_default_volumes +
neutron_metering_agent_extra_volumes }}', 'dimensions':
'{{ neutron_metering_agent_dimensions }}'}, 'ironic-neutron-
agent': {'container_name': 'ironic_neutron_agent', 'image':
'{{ ironic_neutron_agent_image_full }}', 'privileged': False,
'enabled': '{{ enable_ironic_neutron_agent | bool }}', 'group':
'ironic-neutron-agent', 'host_in_groups': \"{{ inventory_hostname
in groups['ironic-neutron-agent'] }}\", 'volumes':
'{{ ironic_neutron_agent_default_volumes +
ironic_neutron_agent_extra_volumes }}', 'dimensions':
'{{ ironic_neutron_agent_dimensions }}'}, 'neutron-tls-proxy':
{'container_name': 'neutron_tls_proxy', 'group': 'neutron-server',
'host_in_groups': \"{{ inventory_hostname in groups['neutron-
server'] }}\", 'enabled': '{{ neutron_enable_tls_backend }}',
'image': '{{ neutron_tls_proxy_image_full }}', 'volumes':
'{{ neutron_tls_proxy_default_volumes +
neutron_tls_proxy_extra_volumes }}', 'dimensions':
'{{ neutron_tls_proxy_dimensions }}', 'healthcheck':
'{{ neutron_tls_proxy_healthcheck }}', 'haproxy':
{'neutron_tls_proxy': {'enabled': '{{ enable_neutron | bool and
neutron_enable_tls_backend | bool }}', 'mode': 'http', 'external':
False, 'port': '{{ neutron_server_port }}', 'listen_port':
'{{ neutron_server_listen_port }}', 'tls_backend': 'yes'},
'neutron_tls_proxy_external': {'enabled': '{{ enable_neutron |
bool and neutron_enable_tls_backend | bool }}', 'mode': 'http',
'external': True, 'port': '{{ neutron_server_port }}',
'listen_port': '{{ neutron_server_listen_port }}', 'tls_backend':
```

```
'yes'}}}}: 'dict object' has no attribute 'neutron-ovn-metadata-
agent'"}
##################################################################

Grund: (Vermutung) falsche ansible-Version
Lösung:
Datei: kolla-ansible/ansible/roles/neutron/defaults/main.yml
#neutron-ovn-metadata-agent:
#  container_name: "neutron_ovn_metadata_agent"
#  image: "{{ neutron_ovn_metadata_agent_image_full }}"
#  privileged: True
#  enabled: "{{ neutron_plugin_agent == 'ovn' }}"
#  host_in_groups: "{{ inventory_hostname in groups['neutron-ovn-
   metadata-agent'] }}"
#  volumes: "{{ neutron_ovn_metadata_agent_default_volumes +
   neutron_ovn_metadata_agent_extra_volumes }}"
#  dimensions: "{{ neutron_ovn_metadata_agent_dimensions }}"
#  healthcheck: "{{ neutron_ovn_metadata_agent_healthcheck }}"

3.) Fehler bei Deployment:
     TASK [mariadb : Check MariaDB service port liveness]
     Fehlermeldung:
##################################################################
fatal: [zead@wally135.cit.tu-berlin.de]: FAILED! => {"changed":
false, "elapsed": 10, "msg": "Timeout when waiting for search
string MariaDB in 130.149.249.145:3306"}
##################################################################
```




```
Grund: Offen.
Lösung: Offen.

4.) Fehler bei deployment:
     RUNNING HANDLER [mariadb : Restart MariaDB container]
     Fehlermeldung:
##################################################################
```

```
fatal: fatal: [zead@wally135.cit.tu-berlin.de]: FAILED! =>
{"changed": true, "msg": "'Traceback (most recent
                          call last):\\n  File
\"/usr/local/lib/python3.6/dist-packages/docker/api/client.py\",
line 222, in _raise_for_status\\n    response.raise_for_status()\\
n  File \"/usr/lib/python3/dist-packages/requests/models.py\",
line 935, in raise_for_status\\n    raise
HTTPError(http_error_msg, response=self)\\
nrequests.exceptions.HTTPError: 404 Client Error: Not Found for
url: http+docker://localunixsocket/v1.41/images/create?
tag=train&fromImage=kolla%2Fubuntu-binary-mariadb-server\\n\\
nDuring handling of the above exception, another exception
occurred:\\n\\nTraceback (most recent call last):\\n  File
\"/tmp/ansible_kolla_docker_payload_pmpbk6iy/ansible_kolla_docker_
payload.zip/ansible/modules/kolla_docker.py\", line 1131, in
main\\n  File
\"/tmp/ansible_kolla_docker_payload_pmpbk6iy/ansible_kolla_docker_
payload.zip/ansible/modules/kolla_docker.py\", line 785, in
recreate_or_restart_container\\n  File
\"/tmp/ansible_kolla_docker_payload_pmpbk6iy/ansible_kolla_docker_
payload.zip/ansible/modules/kolla_docker.py\", line 803, in
start_container\\n  File
\"/tmp/ansible_kolla_docker_payload_pmpbk6iy/ansible_kolla_docker_
payload.zip/ansible/modules/kolla_docker.py\", line 602, in
pull_image\\n  File
\"/usr/local/lib/python3.6/dist-packages/docker/api/image.py\",
line 393, in pull\\n    self._raise_for_status(response)\\n
File
\"/usr/local/lib/python3.6/dist-packages/docker/api/client.py\",
line 224, in _raise_for_status\\n    raise
create_api_error_from_http_exception(e)\\n  File
\"/usr/local/lib/python3.6/dist-packages/docker/errors.py\", line
31, in create_api_error_from_http_exception\\n    raise cls(e,
response=response, explanation=explanation)\\
ndocker.errors.NotFound: 404 Client Error: Not Found (\"manifest
for kolla/ubuntu-binary-mariadb-server:train not found: manifest
unknown: manifest unknown\")\\n'"}
###################################################################
```



Grund: Offen.
Lösung: Offen.

zu 2) Retrospektive
- wollten OpenStack manuell deployen, haben jetzt auf automatik gewechselt (durch starker Empfehlung)
- unser Versuch: mit kolla-ansible & OpenStack-ansible, welches auch immer schneller fertig wird

Konfiguration der Server
Netzwerkinterface
Ansible installiert (,da kolla-ansible das braucht)
docker installiert & eingerichtet
python-Bibliotheken
kolla-ansible installiert & konfiguriert
pip package manager eingerichtet

Fehlt:
kolla-ansible deployment
OpenStack deployment

In allen Nodes:
1. Interfaces in allen Nodes eingerichtet (/etc/netplan/01-netcfg.yaml)
2. Interfaces in /etc/network/interfaces konfiguriert
3. Hosts konfiguriert /etc/hosts → 10.0.42.135: controller, 10.0.42.141: compute1, 10.0.42.142: compute2
4. Kommunikation zwischen den Nodes verifiziert.
5. pip installiert
6. python-pip installiert
7. python-dev
8. libffi-dev

9. gcc
10. libssl-dev
11. chrony installiert und eingerichtet (NTP)
12. /etc/chrony.conf konfiguriert (compute Nodes sehen nur controller und controller erlaubt dies)

In Control Nodes:
1. Ansible installiert
2. kolla-ansible installiert
3. /etc/kolla erstellt
   sudo chown $USER:$USER /etc/kolla
4. /usr/local/share/kolla-ansible/etc_examples/kolla/ in
   /etc/kolla kopiert
   sf/usr/local/share/kolla-ansible/ansible/inventory/*
5. Im Haupt-Verzeichnis: multiple und all-in-node kopiert
   /usr/local/share/kolla-ansible/ansible/inventory/*
6. /etc/ansible/ansible.cfg konfiguriert
7. Inventory: mulptiple konfiguriert (Problem 1)
8. mulptiple nodes Erreichbarkeit verifiziert: bei A → Fehler
   bei B ok
9. Passwort in kolla-ansible erstellt: kolla-genpwd
10. docker installiert
   Kolla globals.yml konfiguriert:
      kolla_base_distro: „Ubunt“
      kolla_install_type: „binary“
      networkt_interface: „eno1“
      neutron_external_interface: „enp2s0“
      kolla_internal_vip_address: „10.0.42.135“
11. bootstrap-servers in multiple ausgeführt (Problem 2)
12. docker konfiguriert
13. verifiziert, ob Docker funktioniert

mulptiple konfiguration: alle Nodes local gesetzt

Problem 1) Wie soll die Verbindung sein (mulptiple)
   A) Ausprobiert wurde SSH, da wird aber ein Passwort aufgefordert
   B) alles nodes „ansible_connection=local§ gesetzt → falsch (Bootstrapping ausprobieren)
Problem 2)