INFN Cloud Use Cases strategy implementation

Marica Antonacci (INFN BA)

Uso e sviluppo di applicazioni e servizi su INFN Cloud (CLueApp)
31 Maggio -1 Giugno 2023

INFN Cloud services implementation strategy

The employed strategy is based on the **Infrastructure as Code paradigm**.

Users describe "**What**" is needed rather than "**How**" a specific service or functionality should be implemented.

The adopted technologies enable a Lego-like approach: services can be composed and modules reused to create the desired infrastructure.

OASIS TOSCA

TOSCA is used to model the topology of the whole application stack

ANSIBLE

Ansible is used to automate the configuration of the virtual environments

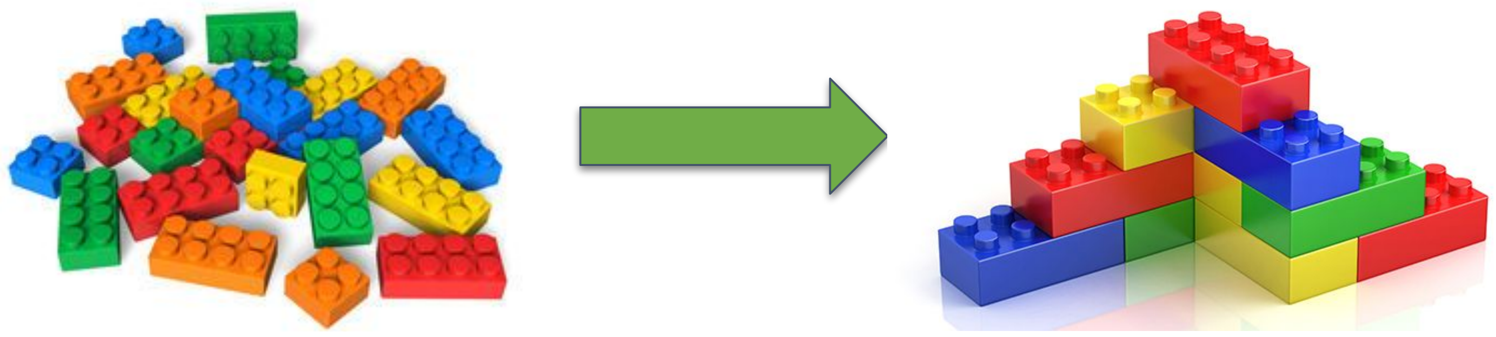
docker

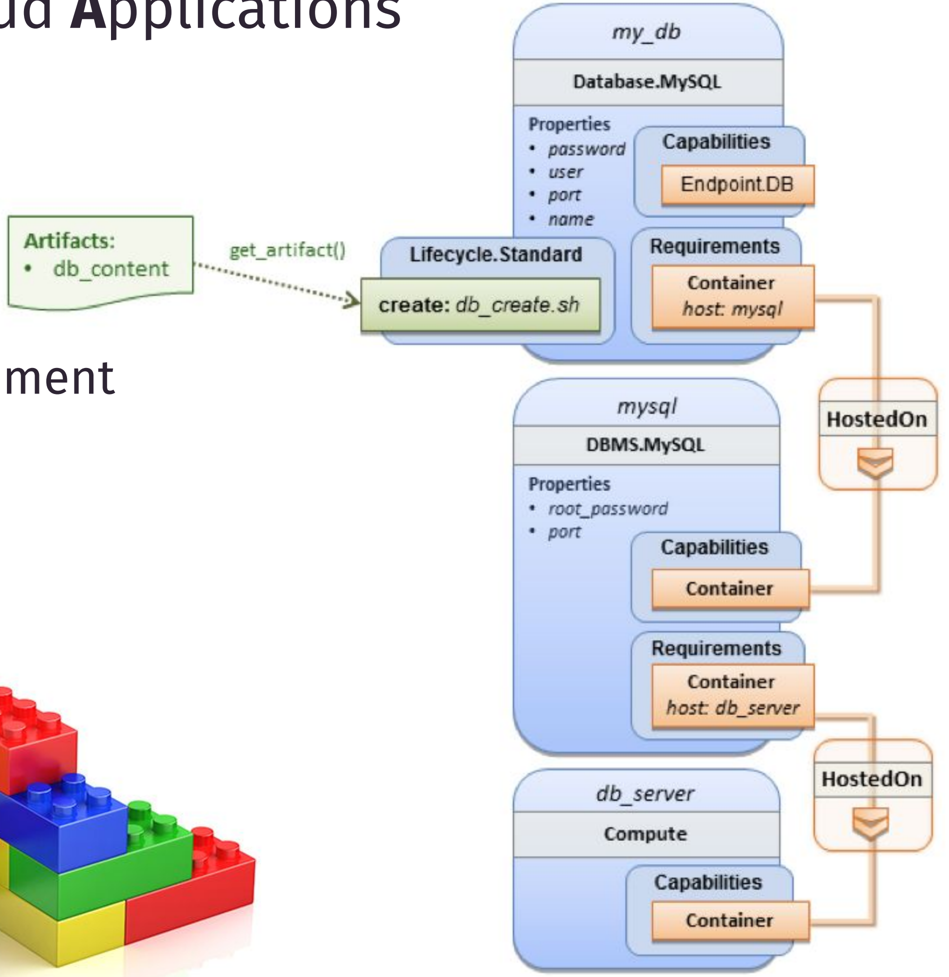
Docker is used to encapsulate the high-level application software and runtime

TOSCA

Topology and **O**rchestration **S**pecification for **C**loud **A**pplications

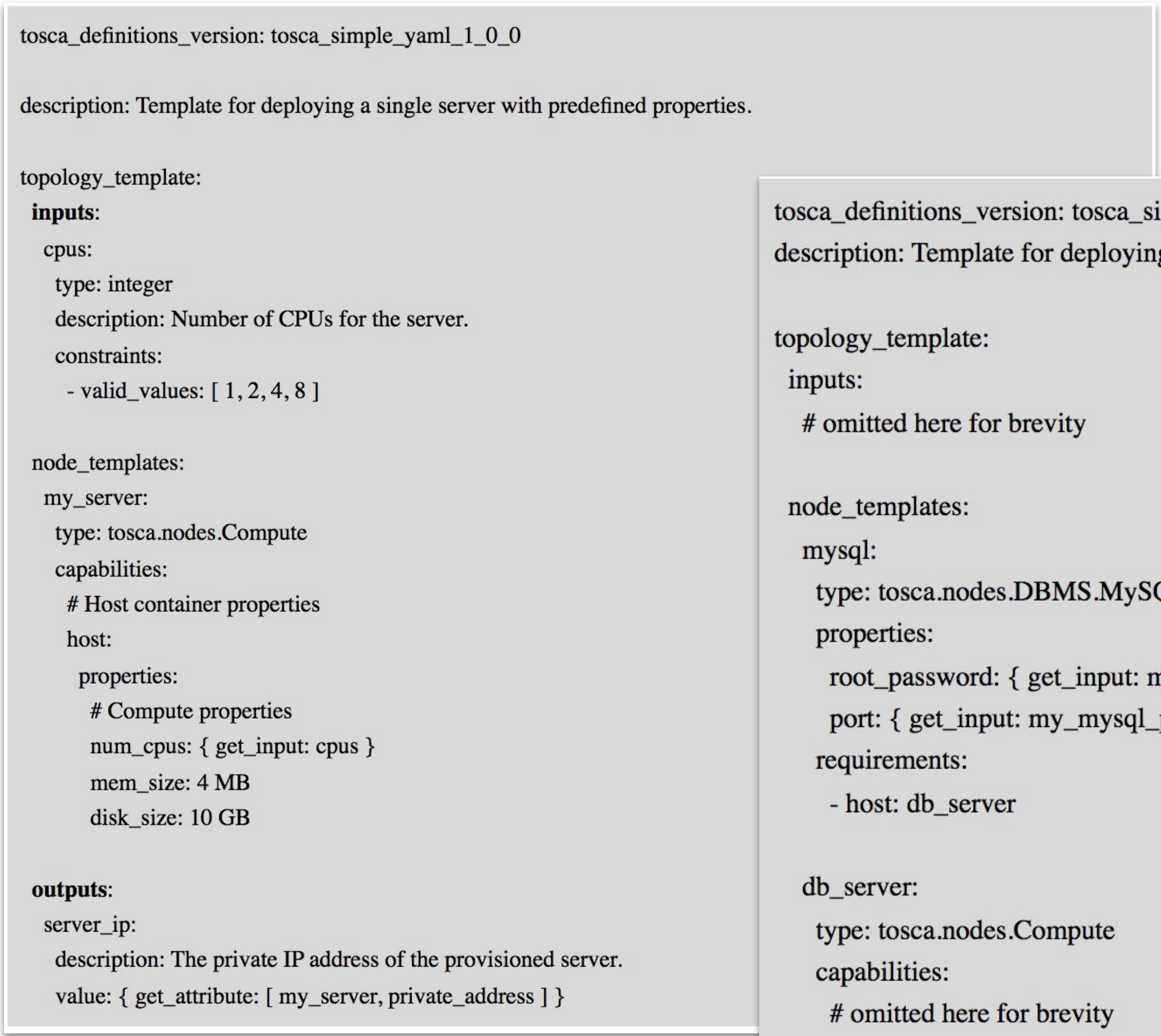
- Goals:
  - Automated Application Deployment and Management.
  - Portability of Application Descriptions and Their Management
  - Interoperability and Reusability of Components



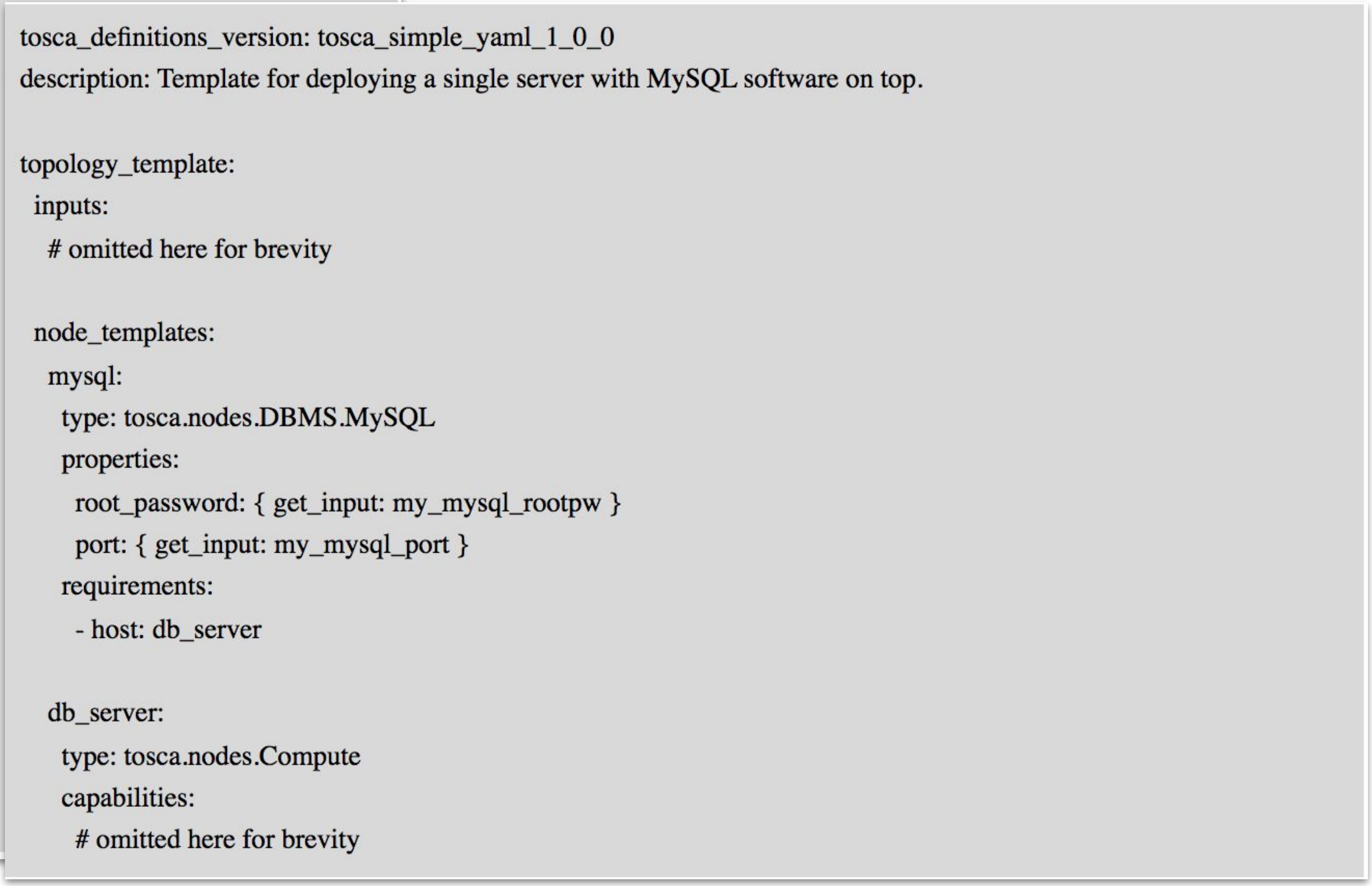


Ref: TOSCA Simple Profile in YAML Version 1.1

Template example

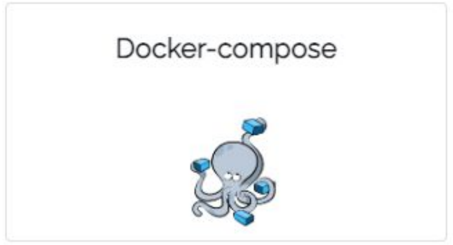


```
tosca_definitions_version: tosca_simple_yaml_1_0_0

description: Template for deploying a single server with predefined properties.

topology_template:
 inputs:
  cpus:
   type: integer
   description: Number of CPUs for the server.
   constraints:
    - valid_values: [ 1, 2, 4, 8 ]

 node_templates:
  my_server:
   type: tosca.nodes.Compute
   capabilities:
    # Host container properties
    host:
     properties:
      # Compute properties
      num_cpus: { get_input: cpus }
      mem_size: 4 MB
      disk_size: 10 GB

 outputs:
  server_ip:
   description: The private IP address of the provisioned server.
   value: { get_attribute: [ my_server, private_address ] }
```



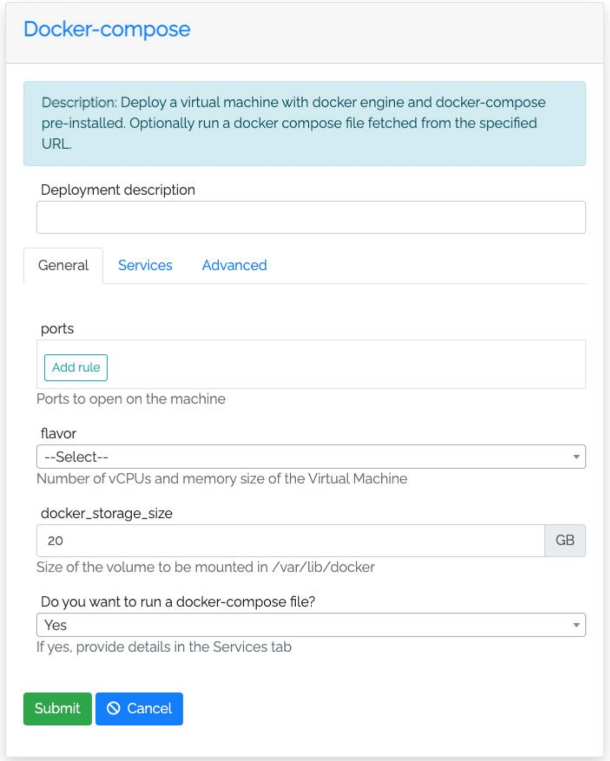
```
tosca_definitions_version: tosca_simple_yaml_1_0_0
description: Template for deploying a single server with MySQL software on top.

topology_template:
 inputs:
  # omitted here for brevity

 node_templates:
  mysql:
   type: tosca.nodes.DBMS.MySQL
   properties:
    root_password: { get_input: my_mysql_rootpw }
    port: { get_input: my_mysql_port }
   requirements:
    - host: db_server

  db_server:
   type: tosca.nodes.Compute
   capabilities:
    # omitted here for brevity
```

Docker compose base implementation



Let's have a look at the TOSCA template

https://baltig.infn.it/infn-cloud/tosca-templates/-/blob/master/docker/docker_compose.yaml



Docker-compose

Description: Deploy a virtual machine with docker engine and docker-compose pre-installed. Optionally run a docker compose file fetched from the specified URL.

Deployment description

General | Services | Advanced

ports

Add rule

Ports to open on the machine

flavor

--Select--

Number of vCPUs and memory size of the Virtual Machine

docker_storage_size

20 GB

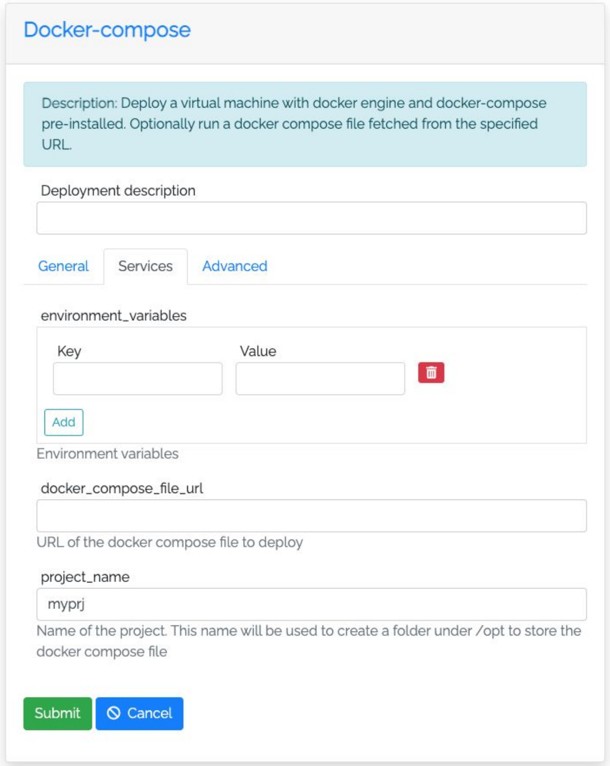
Size of the volume to be mounted in /var/lib/docker

Do you want to run a docker-compose file?

Yes

If yes, provide details in the Services tab

Submit | Cancel



Docker-compose

Description: Deploy a virtual machine with docker engine and docker-compose pre-installed. Optionally run a docker compose file fetched from the specified URL.

Deployment description

General | Services | Advanced

environment_variables

Key	Value

Add

Environment variables

docker_compose_file_url

URL of the docker compose file to deploy

project_name

myprj

Name of the project. This name will be used to create a folder under /opt to store the docker compose file

Submit | Cancel

TOSCA definition

```
docker_compose_service:
  type: tosca.nodes.indigo.DockerCompose
  properties:
    project_name:  { get_input: project_name }
    docker_compose_file_url: { get_input: docker_compose_file_url }
    environment_variables: { get_input: environment_variables }
  requirements:
    - host: server

server:
  type: tosca.nodes.indigo.Compute
  properties:
    os_users: { get_input: users }
  capabilities:
    endpoint:
      properties:
        ports: { get_input: service_ports }
    host:
      properties:
        num_cpus: { get_input: num_cpus }
        mem_size: { get_input: mem_size }
    os:
      properties:
        distribution: ubuntu
        type: linux
        version: 20.04
```

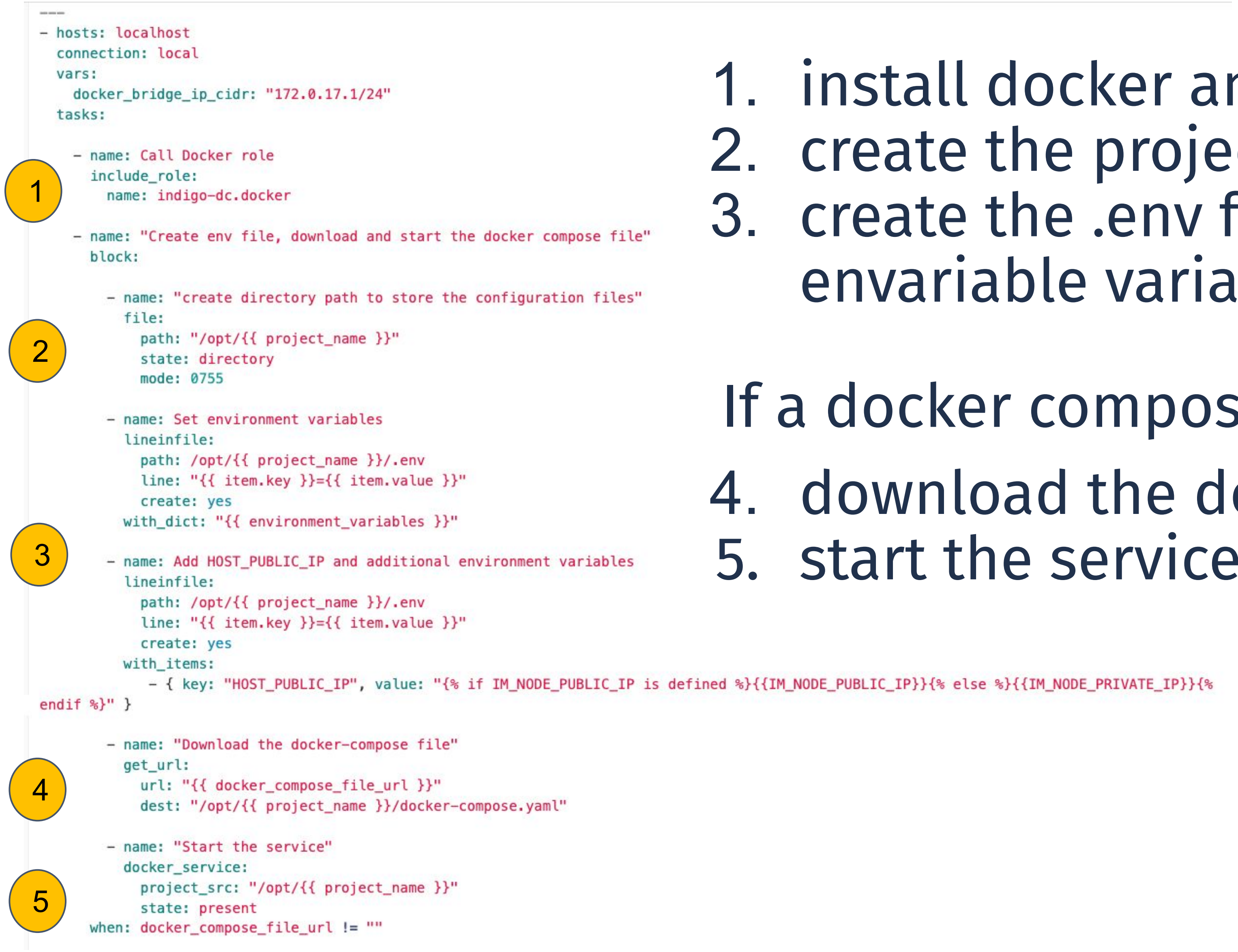
```
tosca.nodes.indigo.DockerCompose:
  derived_from: tosca.nodes.SoftwareComponent
  properties:
    docker_compose_version:
      type: version
      required: no
      default: 1.25.5
    docker_compose_file_url:
      type: string
      required: no
      default: ""
    environment_variables:
      required: no
      default: []
      type: list
      entry_schema:
        type: map
        entry_schema:
          type: string
    project_name:
      type: string
      required: yes
  artifacts:
    docker_role:
      file: indigo-dc.docker,v2.1.3
      type: tosca.artifacts.AnsibleGalaxy.role
  interfaces:
    Standard:
      start:
        implementation: https://baltig.infn.it/infn-cloud/tosca-types/raw/master/artifacts/docker/docker-compose_start.yml
        inputs:
          docker_compose_version: { get_property: [ SELF, docker_compose_version ] }
          docker_compose_file_url:  { get_property: [ SELF, docker_compose_file_url ] }
          project_name:  { get_property: [ SELF, project_name ] }
          environment_variables: { get_property: [ SELF, environment_variables ] }
```

Ansible role

Ansible playbook

https://baltig.infn.it/infn-cloud/tosca-types/-/blob/master/tosca_types/infrastructure/docker_types.yaml

The playbook



```
---
- hosts: localhost
  connection: local
  vars:
    docker_bridge_ip_cidr: "172.0.17.1/24"
  tasks:

    - name: Call Docker role
      include_role:
        name: indigo-dc.docker

    - name: "Create env file, download and start the docker compose file"
      block:

        - name: "create directory path to store the configuration files"
          file:
            path: "/opt/{{ project_name }}"
            state: directory
            mode: 0755

        - name: Set environment variables
          lineinfile:
            path: /opt/{{ project_name }}/.env
            line: "{{ item.key }}={{ item.value }}"
            create: yes
          with_dict: "{{ environment_variables }}"

        - name: Add HOST_PUBLIC_IP and additional environment variables
          lineinfile:
            path: /opt/{{ project_name }}/.env
            line: "{{ item.key }}={{ item.value }}"
            create: yes
          with_items:
            - { key: "HOST_PUBLIC_IP", value: "{% if IM_NODE_PUBLIC_IP is defined %}{{IM_NODE_PUBLIC_IP}}{% else %}{{IM_NODE_PRIVATE_IP}}{%
endif %}" }

        - name: "Download the docker-compose file"
          get_url:
            url: "{{ docker_compose_file_url }}"
            dest: "/opt/{{ project_name }}/docker-compose.yaml"

        - name: "Start the service"
          docker_service:
            project_src: "/opt/{{ project_name }}"
            state: present
      when: docker_compose_file_url != ""
```

1. install docker and compose
2. create the project dir
3. create the .env file with all the envariable variables

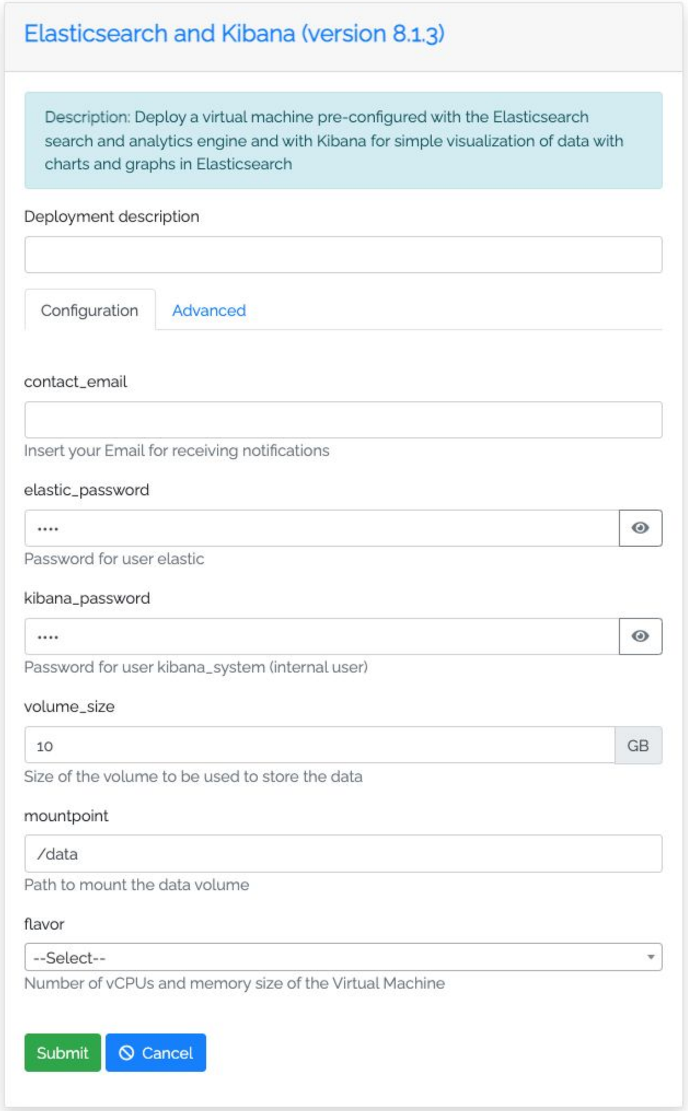
If a docker compose file url is defined:

4. download the docker compose file
5. start the services

EK services implementation

The elasticsearch + kibana (EK) service has been implemented extending the basic docker compose service, deriving the custom type from ***tosca.nodes.indigo.DockerCompose***

EK service implementation



Elasticsearch and Kibana (version 8.1.3)

Description: Deploy a virtual machine pre-configured with the Elasticsearch search and analytics engine and with Kibana for simple visualization of data with charts and graphs in Elasticsearch

Deployment description

Configuration | Advanced

contact_email

Insert your Email for receiving notifications

elastic_password

····

Password for user elastic

kibana_password

····

Password for user kibana_system (internal user)

volume_size

10 GB

Size of the volume to be used to store the data

mountpoint

/data

Path to mount the data volume

flavor

--Select--

Number of vCPUs and memory size of the Virtual Machine

Submit | Cancel

TOSCA template:

https://baltig.infn.it/infn-cloud/tosca-templates/-/blob/master/single-vm/elasticsearch_kibana.yaml

```
docker_compose_service:
  type: tosca.nodes.indigo.DockerCompose.Elastic
  properties:
    project_name: elastic
    environment_variables:
      - ELASTIC_VERSION: "8.1.3"
      - ELASTIC_PASSWORD: { get_input: elastic_password }
      - KIBANA_PASSWORD: { get_input: kibana_password }
      - CERT_EMAIL: { get_input: contact_email }
      - DATA_DIR: { get_input: mountpoint }
  requirements:
    - host: kibana_es_server
```

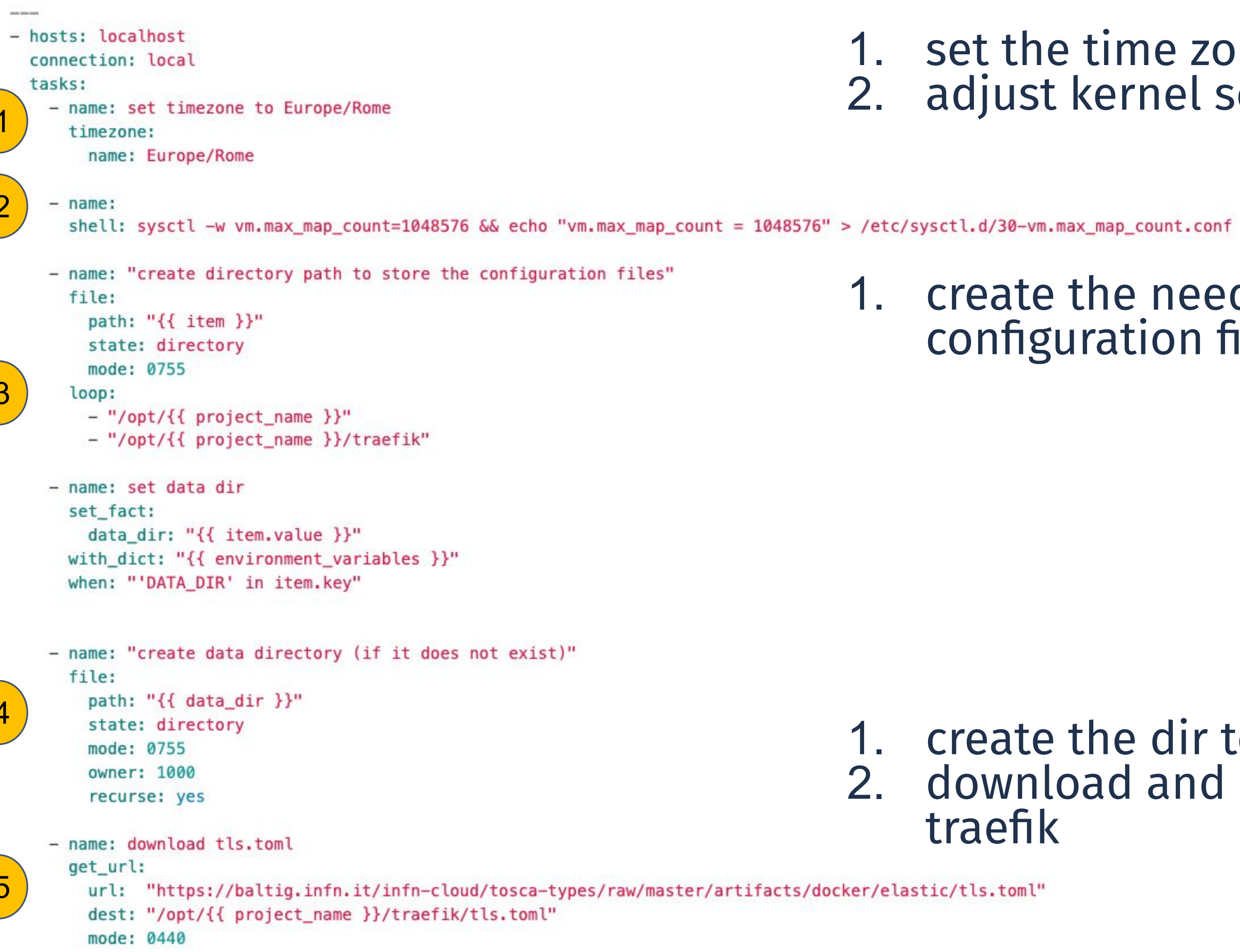
Derived type

```
tosca.nodes.indigo.DockerCompose.Elastic:
  derived_from: tosca.nodes.indigo.DockerCompose
  properties:
    docker_compose_file_url:
      type: string
      default: https://baltig.infn.it/infn-cloud/tosca-types/raw/master/artifacts/docker/elastic/docker-compose.yml
  artifacts:
    docker_role:
      file: indigo-dc.docker,v2.1.3
      type: tosca.artifacts.AnsibleGalaxy.role
  interfaces:
    Standard:
      configure:
        implementation: https://baltig.infn.it/infn-cloud/tosca-types/raw/master/artifacts/docker/elastic/configure.yml
        inputs:
          project_name:  { get_property: [ SELF, project_name ] }
          environment_variables: { get_property: [ SELF, environment_variables ] }
      start:
        implementation: https://baltig.infn.it/infn-cloud/tosca-types/raw/master/artifacts/docker/docker-compose_start.yml
        inputs:
          docker_compose_version: { get_property: [ SELF, docker_compose_version ] }
          docker_compose_file_url:  { get_property: [ SELF, docker_compose_file_url ] }
          project_name:  { get_property: [ SELF, project_name ] }
```

The property *docker_compose_file_url* is overridden providing the default docker compose file. All other properties are inherited by the parent type

The interfaces are specialised too in order to perform custom preliminary configurations (see next slide)

Customized playbook



```yaml
---
- hosts: localhost
  connection: local
  tasks:
```

1

```yaml
    - name: set timezone to Europe/Rome
      timezone:
        name: Europe/Rome
```

2

```yaml
    - name:
      shell: sysctl -w vm.max_map_count=1048576 && echo "vm.max_map_count = 1048576" > /etc/sysctl.d/30-vm.max_map_count.conf
```

3

```yaml
    - name: "create directory path to store the configuration files"
      file:
        path: "{{ item }}"
        state: directory
        mode: 0755
      loop:
        - "/opt/{{ project_name }}"
        - "/opt/{{ project_name }}/traefik"

    - name: set data dir
      set_fact:
        data_dir: "{{ item.value }}"
      with_dict: "{{ environment_variables }}"
      when: "'DATA_DIR' in item.key"
```

4

```yaml
    - name: "create data directory (if it does not exist)"
      file:
        path: "{{ data_dir }}"
        state: directory
        mode: 0755
        owner: 1000
        recurse: yes
```

5

```yaml
    - name: download tls.toml
      get_url:
        url:  "https://baltig.infn.it/infn-cloud/tosca-types/raw/master/artifacts/docker/elastic/tls.toml"
        dest: "/opt/{{ project_name }}/traefik/tls.toml"
        mode: 0440
```

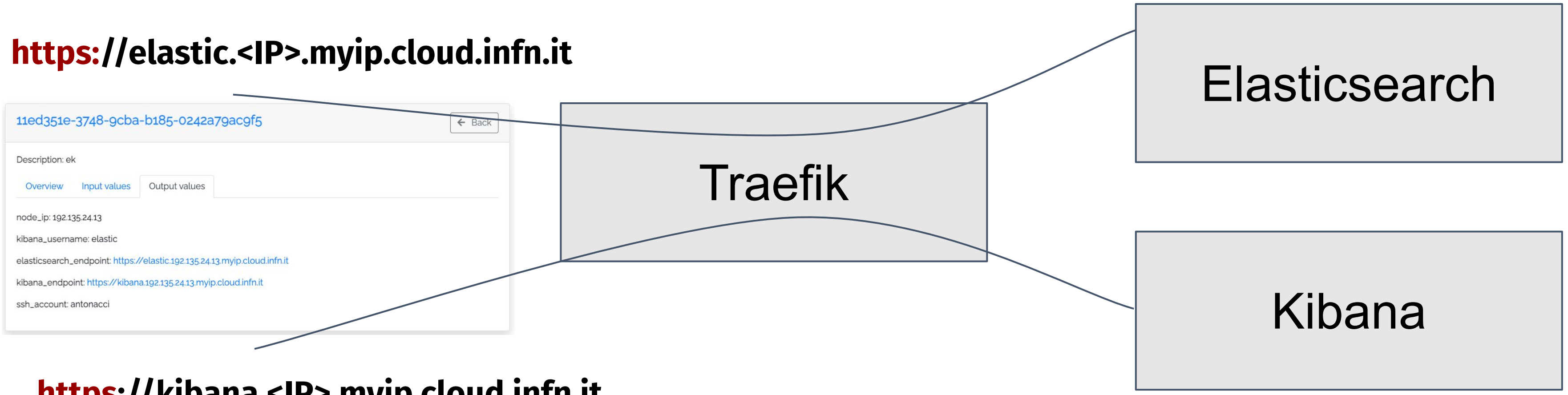
1. set the time zone
2. adjust kernel settings (see doc)

1. create the needed dirs to host configuration files

1. create the dir to store the collected data
2. download and install the TLS settings for traefik

The docker compose file

https://elastic.<IP>.myip.cloud.infn.it



11ed351e-3748-9cba-b185-0242a79ac9f5

Description: ek

node_ip: 192.135.24.13

kibana_username: elastic

elasticsearch_endpoint: https://elastic.192.135.24.13.myip.cloud.infn.it

kibana_endpoint: https://kibana.192.135.24.13.myip.cloud.infn.it

ssh_account: antonacci

https://kibana.<IP>.myip.cloud.infn.it

Traefik terminates the SSL connections: it is configured to use an ACME provider (Let's Encrypt) for automatic certificate generation.

https://baltig.infn.it/infn-cloud/tosca-types/-/blob/master/artifacts/docker/elastic/docker-compose.yml

Sync&Share service

The INFN-Cloud Sync&Share aaS is currently based on the popular ***ownCloud/Nextcloud*** storage solution.
INFN-Cloud users have full control over the configuration parameters of their Cloud Storage instance, as well as on third party accesses to the stored data.
Main features:

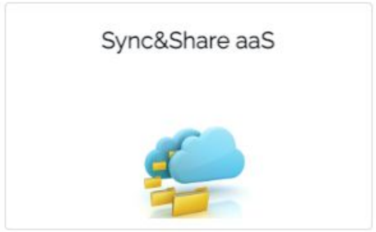
- **S3 based Object Storage backend** where data is replicated over two backbone data centers (CNAF, BARI)
- Authentication/Authorization based on **INFN-Cloud IAM** (via OIDC)
- **programmatic access** to user data via Rclone, including remote mount and folder sync
- embedded, automated DB and configuration **backup**
- embedded, pre-configured **monitoring system** with alert notifications

Service implementation

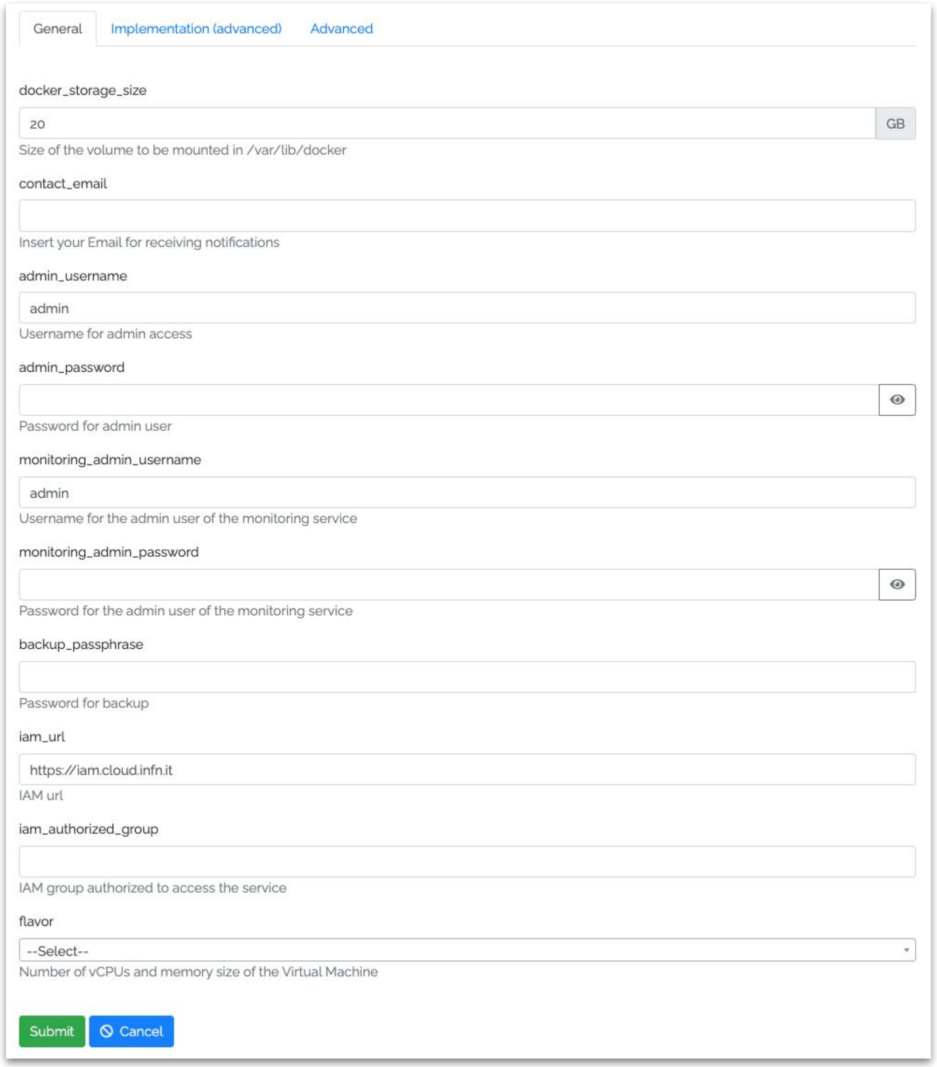
The core setup is based on a docker compose file like the EK service, but in this case the implementation is different.

Since the configuration of this service is a bit more complex, we decided not to derive from the tosca.nodes.indigo.DockerCompose.

A new tosca type has been developed as a new Software Component.



Service configuration and deployment outputs



docker_storage_size
20 GB
Size of the volume to be mounted in /var/lib/docker

contact_email

Insert your Email for receiving notifications

admin_username
admin
Username for admin access

admin_password

Password for admin user

monitoring_admin_username
admin
Username for the admin user of the monitoring service

monitoring_admin_password

Password for the admin user of the monitoring service

backup_passphrase

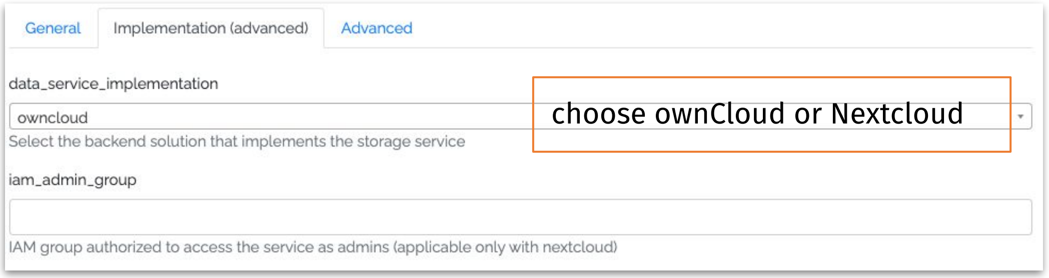
Password for backup

iam_url
https://iam.cloud.infn.it
IAM url

iam_authorized_group

IAM group authorized to access the service

flavor
--Select--
Number of vCPUs and memory size of the Virtual Machine

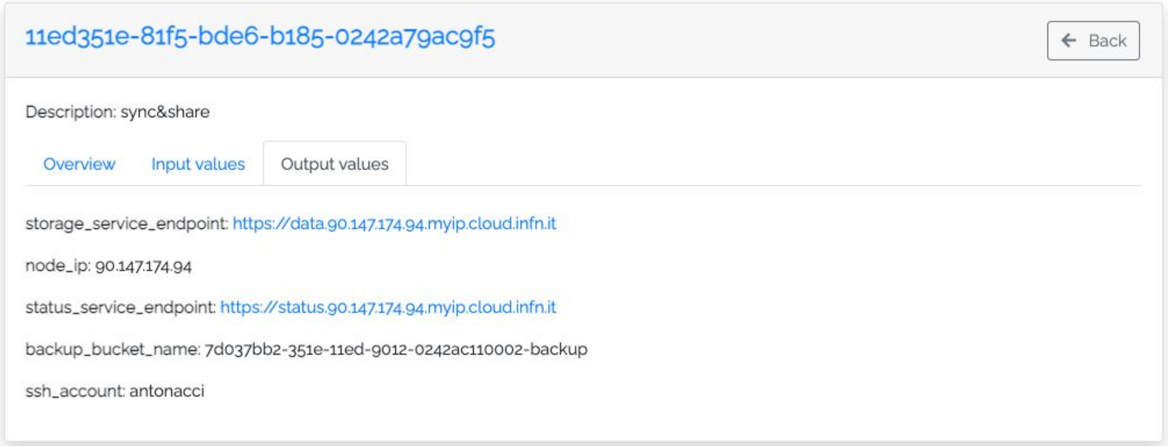


data_service_implementation
owncloud
Select the backend solution that implements the storage service

choose ownCloud or Nextcloud

iam_admin_group

IAM group authorized to access the service as admins (applicable only with nextcloud)



Description: sync&share

storage_service_endpoint: https://data.90.147.174.94.myip.cloud.infn.it

node_ip: 90.147.174.94

status_service_endpoint: https://status.90.147.174.94.myip.cloud.infn.it

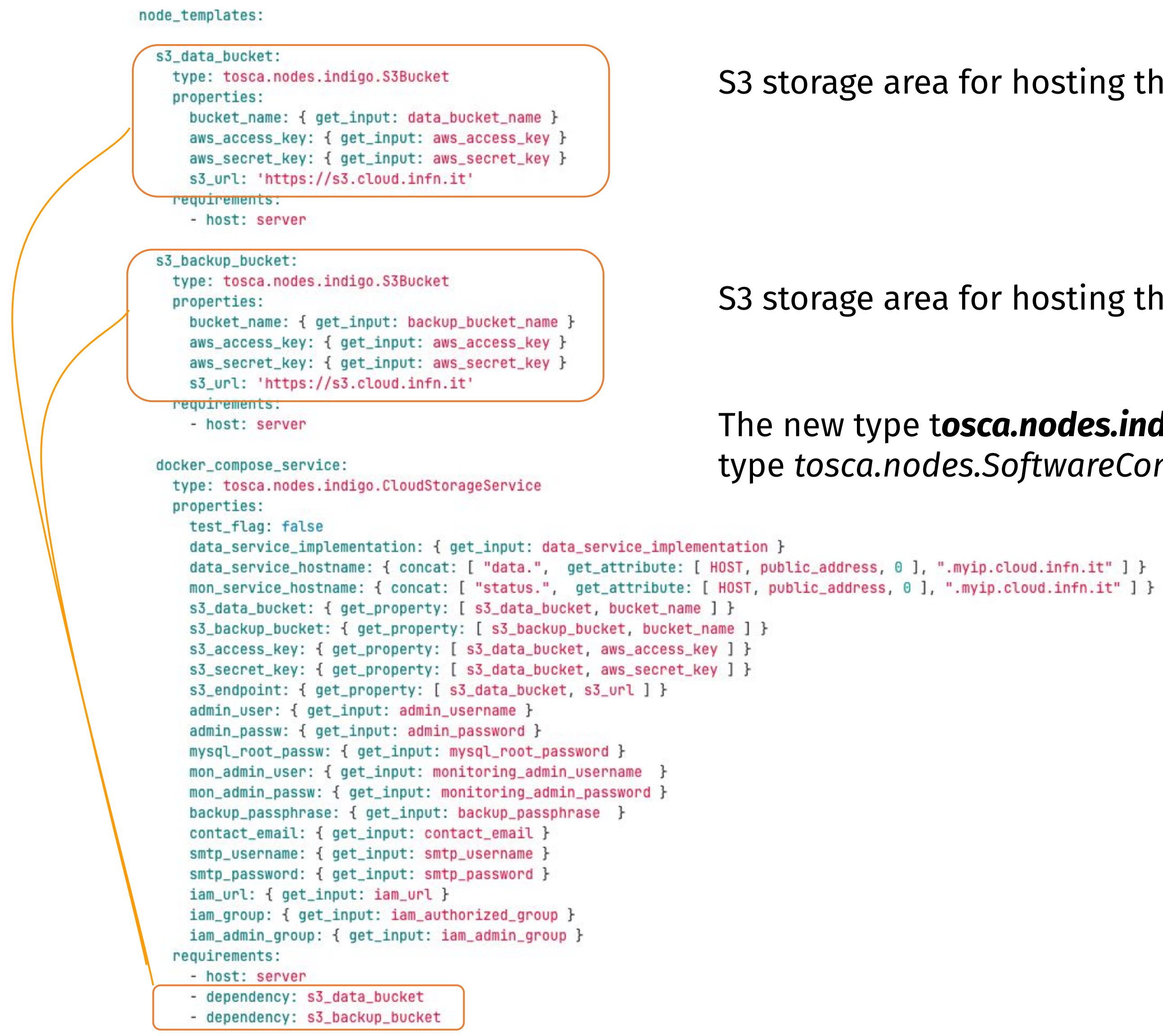
backup_bucket_name: 7d037bb2-351e-11ed-9012-0242ac110002-backup

ssh_account: antonacci

TOSCA template:

https://baltig.infn.it/infn-cloud/tosca-templates/-/blob/master/single-vm/cloud_storage_service.yaml

TOSCA definition



```
node_templates:

  s3_data_bucket:
    type: tosca.nodes.indigo.S3Bucket
    properties:
      bucket_name: { get_input: data_bucket_name }
      aws_access_key: { get_input: aws_access_key }
      aws_secret_key: { get_input: aws_secret_key }
      s3_url: 'https://s3.cloud.infn.it'
    requirements:
      - host: server

  s3_backup_bucket:
    type: tosca.nodes.indigo.S3Bucket
    properties:
      bucket_name: { get_input: backup_bucket_name }
      aws_access_key: { get_input: aws_access_key }
      aws_secret_key: { get_input: aws_secret_key }
      s3_url: 'https://s3.cloud.infn.it'
    requirements:
      - host: server

  docker_compose_service:
    type: tosca.nodes.indigo.CloudStorageService
    properties:
      test_flag: false
      data_service_implementation: { get_input: data_service_implementation }
      data_service_hostname: { concat: [ "data.",  get_attribute: [ HOST, public_address, 0 ], ".myip.cloud.infn.it" ] }
      mon_service_hostname: { concat: [ "status.",  get_attribute: [ HOST, public_address, 0 ], ".myip.cloud.infn.it" ] }
      s3_data_bucket: { get_property: [ s3_data_bucket, bucket_name ] }
      s3_backup_bucket: { get_property: [ s3_backup_bucket, bucket_name ] }
      s3_access_key: { get_property: [ s3_data_bucket, aws_access_key ] }
      s3_secret_key: { get_property: [ s3_data_bucket, aws_secret_key ] }
      s3_endpoint: { get_property: [ s3_data_bucket, s3_url ] }
      admin_user: { get_input: admin_username }
      admin_passw: { get_input: admin_password }
      mysql_root_passw: { get_input: mysql_root_password }
      mon_admin_user: { get_input: monitoring_admin_username  }
      mon_admin_passw: { get_input: monitoring_admin_password }
      backup_passphrase: { get_input: backup_passphrase  }
      contact_email: { get_input: contact_email }
      smtp_username: { get_input: smtp_username }
      smtp_password: { get_input: smtp_password }
      iam_url: { get_input: iam_url }
      iam_group: { get_input: iam_authorized_group }
      iam_admin_group: { get_input: iam_admin_group }
    requirements:
      - host: server
      - dependency: s3_data_bucket
      - dependency: s3_backup_bucket
```

S3 storage area for hosting the user data

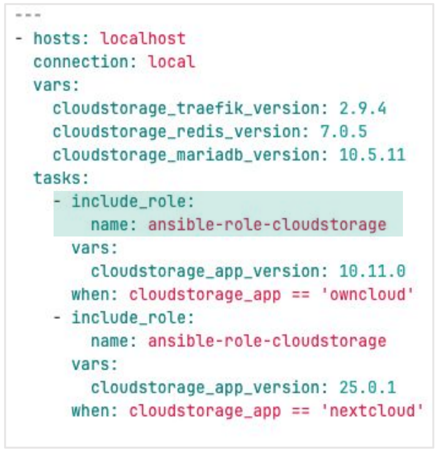
S3 storage area for hosting the backup data

The new type t***osca.nodes.indigo.CloudStorageService*** is derived from the normative type *tosca.nodes.SoftwareComponent*

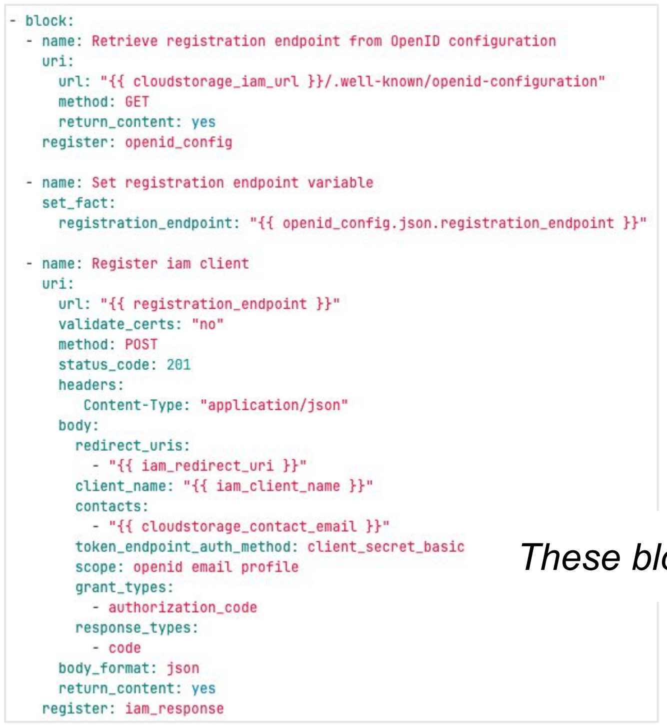
The ansible playbook

https://baltig.infn.it/infn-cloud/tosca-types/raw/master/artifacts/cloudstorage/configure.yml

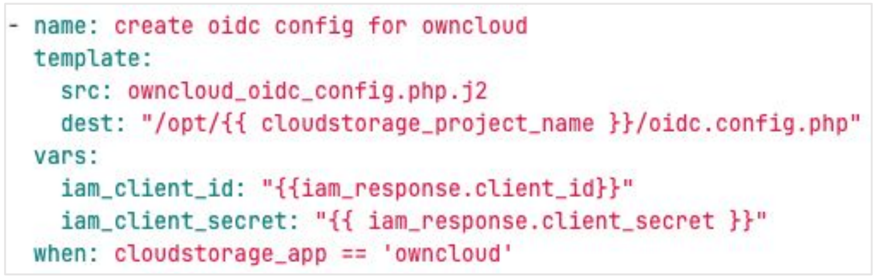
role repository: https://baltig.infn.it/infn-cloud/ansible-role-cloudstorage



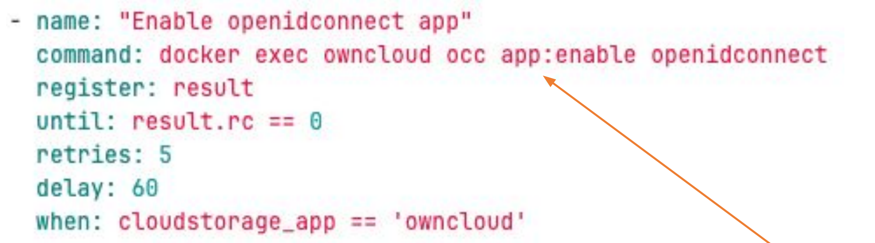
```yaml
---
- hosts: localhost
  connection: local
  vars:
    cloudstorage_traefik_version: 2.9.4
    cloudstorage_redis_version: 7.0.5
    cloudstorage_mariadb_version: 10.5.11
  tasks:
    - include_role:
        name: ansible-role-cloudstorage
      vars:
        cloudstorage_app_version: 10.11.0
      when: cloudstorage_app == 'owncloud'
    - include_role:
        name: ansible-role-cloudstorage
      vars:
        cloudstorage_app_version: 25.0.1
      when: cloudstorage_app == 'nextcloud'
```



```yaml
- block:
  - name: Retrieve registration endpoint from OpenID configuration
    uri:
      url: "{{ cloudstorage_iam_url }}/.well-known/openid-configuration"
      method: GET
      return_content: yes
    register: openid_config

  - name: Set registration endpoint variable
    set_fact:
      registration_endpoint: "{{ openid_config.json.registration_endpoint }}"

  - name: Register iam client
    uri:
      url: "{{ registration_endpoint }}"
      validate_certs: "no"
      method: POST
      status_code: 201
      headers:
         Content-Type: "application/json"
      body:
        redirect_uris:
          - "{{ iam_redirect_uri }}"
        client_name: "{{ iam_client_name }}"
        contacts:
          - "{{ cloudstorage_contact_email }}"
        token_endpoint_auth_method: client_secret_basic
        scope: openid email profile
        grant_types:
          - authorization_code
        response_types:
          - code
      body_format: json
      return_content: yes
    register: iam_response
```



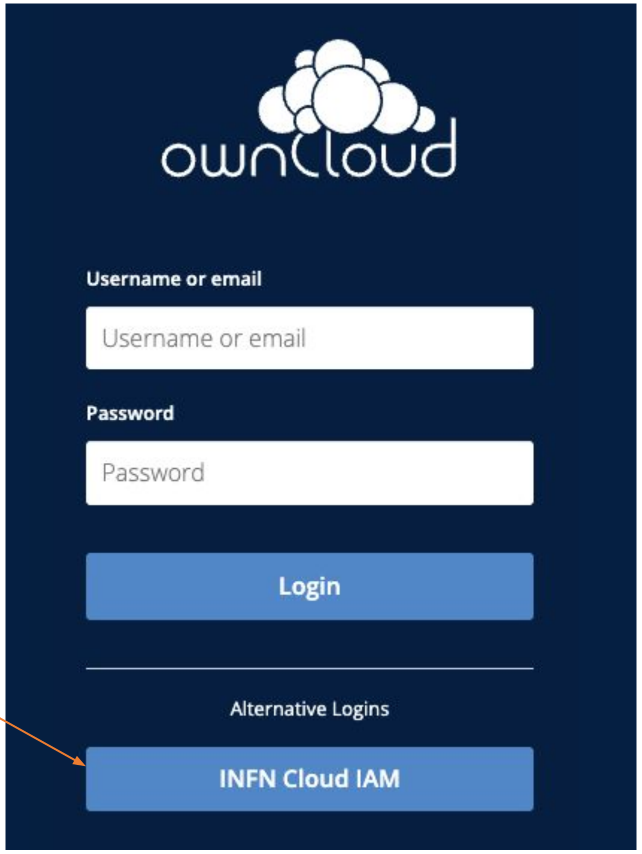
```yaml
- name: create oidc config for owncloud
  template:
    src: owncloud_oidc_config.php.j2
    dest: "/opt/{{ cloudstorage_project_name }}/oidc.config.php"
  vars:
    iam_client_id: "{{iam_response.client_id}}"
    iam_client_secret: "{{ iam_response.client_secret }}"
  when: cloudstorage_app == 'owncloud'
```



```yaml
- name: "Enable openidconnect app"
  command: docker exec owncloud occ app:enable openidconnect
  register: result
  until: result.rc == 0
  retries: 5
  delay: 60
  when: cloudstorage_app == 'owncloud'
```

Once the services are up and running, the oidc app is enabled (the configuration for nextcloud is similar)

These blocks of tasks configure the integration with IAM



The docker compose file (template)

https://baltig.infn.it/infn-cloud/ansible-role-cloudstorage/-/blob/main/templates/docker-compose.yaml

```
services:
  proxy:
    container_name: proxy
    image: harbor.cloud.infn.it/cache/library/traefik:${TRAEFIK_VERSION}
```

```
  {% if cloudstorage_app == 'nextcloud' %}
  {% include 'nextcloud.j2' %}
  {% elif cloudstorage_app == 'owncloud' %}
  {% include 'owncloud.j2' %}
  {% endif %}
```

```
nextcloud:
  container_name: nextcloud
  image: harbor.cloud.infn.it/cache/library/nextcloud:${APPCLOUD_VERSION}
```

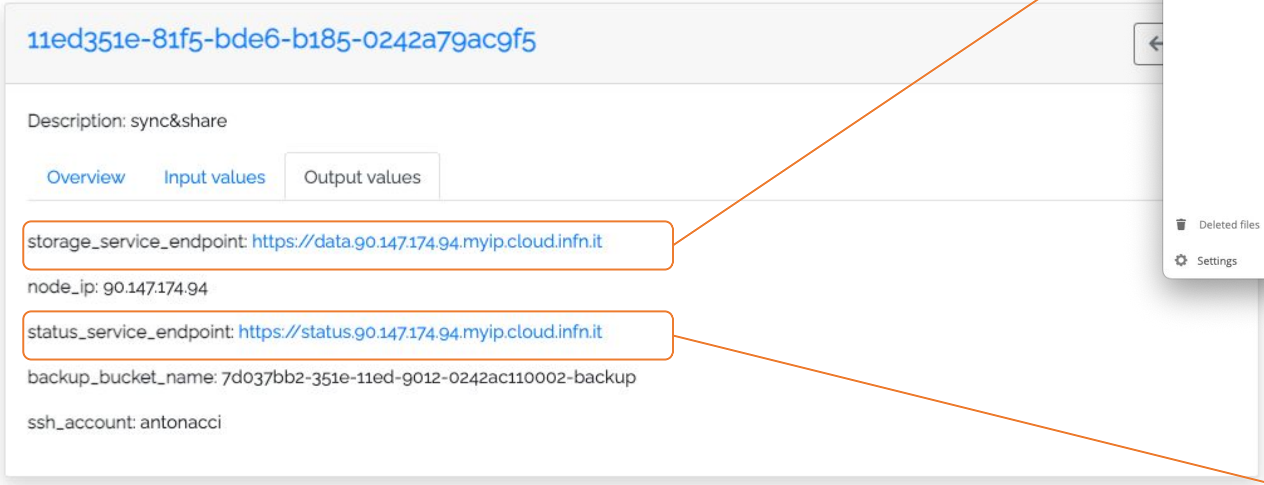
```
owncloud:
  container_name: owncloud
  image: harbor.cloud.infn.it/cache/owncloud/server:${APPCLOUD_VERSION}
```

```
  redis:
    image: harbor.cloud.infn.it/cache/library/redis:${REDIS_VERSION}

  db:
    container_name: db
    image: harbor.cloud.infn.it/cache/library/mariadb:${MARIADB_VERSION}

  nagios:
    container_name: nagios
    image: harbor.cloud.infn.it/library/storageservice-nagios:1.2
```

```
backup:
  image: harbor.cloud.infn.it/library/storageservice-backup
  container_name: backup
  volumes:
    - files:/backup/files
    - backup:/backup/db
    - backup-logs:/backup/logs
    - nagios-conf:/backup/nagios/conf
    - nagios-data:/backup/nagios/data
    - letsencrypt:/backup/letsencrypt
    - /etc/timezone:/etc/timezone:ro
    - /etc/localtime:/etc/localtime:ro
    - duplicity-metadata:/duplicity-metadata
```

```
dbbackup:
  image: harbor.cloud.infn.it/library/storageservice-mariadb-backup
  container_name: dbbackup
  volumes:
    - mariadb:/var/lib/mysql
    - backup:/backup
    - /etc/timezone:/etc/timezone:ro
    - /etc/localtime:/etc/localtime:ro
  environment:
    - CRON_SCHEDULE=10 15 * * *
```

Services are accessed through the reverse proxy based on Traefik



11ed351e-81f5-bde6-b185-0242a79ac9f5

Description: sync&share

storage_service_endpoint: https://data.90.147.174.94.myip.cloud.infn.it

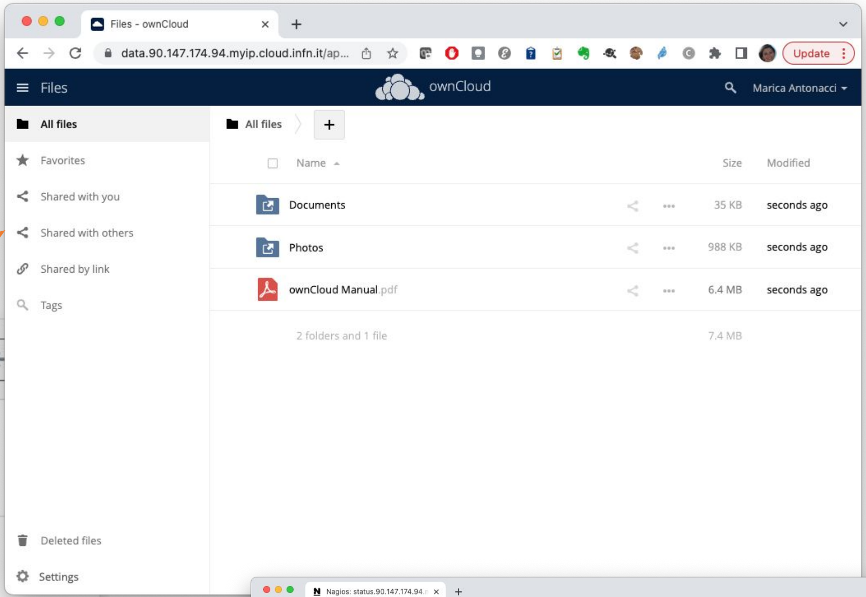
node_ip: 90.147.174.94

status_service_endpoint: https://status.90.147.174.94.myip.cloud.infn.it

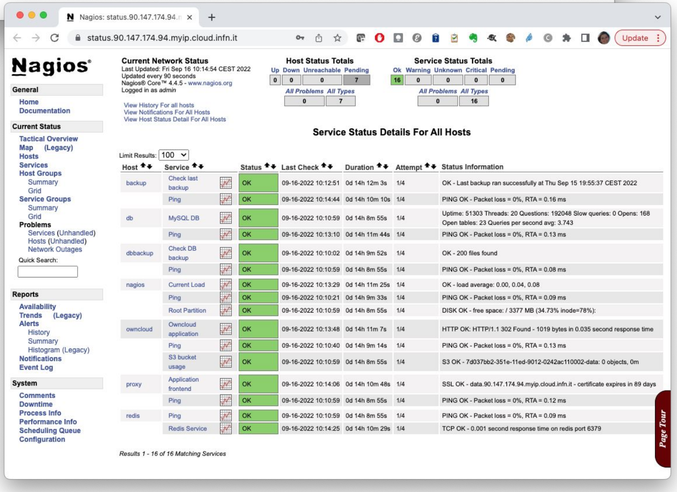
backup_bucket_name: 7d037bb2-351e-11ed-9012-0242ac110002-backup

ssh_account: antonacci

https://data.<IP>.myip.cloud.infn.it



Name	Size	Modified
Documents	35 KB	seconds ago
Photos	988 KB	seconds ago
ownCloud Manual.pdf	6.4 MB	seconds ago
2 folders and 1 file	7.4 MB	



Current Network Status
Last Updated: Fri Sep 16 10:14:54 CEST 2022
Updated every 90 seconds
Nagios® Core™ 4.4.5 - www.nagios.org
Logged in as *admin*

View History For all hosts
View Notifications For All Hosts
View Host Status Detail For All Hosts

Host Status Totals

Up	Down	Unreachable	Pending
0	0	0	7

All Problems	All Types
0	7

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
16	0	0	0	0

All Problems	All Types
0	16

Service Status Details For All Hosts

Limit Results: 100

Host	Service	Status	Last Check	Duration	Attempt	Status Information
backup	Check last backup	OK	09-16-2022 10:12:51	0d 14h 12m 3s	1/4	OK - Last backup ran successfully at Thu Sep 15 19:55:37 CEST 2022
	Ping	OK	09-16-2022 10:14:44	0d 14h 10m 10s	1/4	PING OK - Packet loss = 0%, RTA = 0.16 ms
db	MySQL DB	OK	09-16-2022 10:10:59	0d 14h 8m 55s	1/4	Uptime: 51303 Threads: 20 Questions: 192048 Slow queries: 0 Opens: 168 Open tables: 23 Queries per second avg: 3.743
	Ping	OK	09-16-2022 10:13:10	0d 14h 11m 44s	1/4	PING OK - Packet loss = 0%, RTA = 0.13 ms
dbbackup	Check DB backup	OK	09-16-2022 10:10:02	0d 14h 9m 52s	1/4	OK - 200 files found
	Ping	OK	09-16-2022 10:10:59	0d 14h 8m 55s	1/4	PING OK - Packet loss = 0%, RTA = 0.08 ms
nagios	Current Load	OK	09-16-2022 10:13:29	0d 14h 11m 25s	1/4	OK - load average: 0.00, 0.04, 0.08
	Ping	OK	09-16-2022 10:10:21	0d 14h 9m 33s	1/4	PING OK - Packet loss = 0%, RTA = 0.09 ms
	Root Partition	OK	09-16-2022 10:10:59	0d 14h 8m 55s	1/4	DISK OK - free space: / 3377 MB (34.73% inode=78%):
owncloud	Owncloud application	OK	09-16-2022 10:13:48	0d 14h 11m 7s	1/4	HTTP OK: HTTP/1.1 302 Found - 1019 bytes in 0.035 second response time
	Ping	OK	09-16-2022 10:10:40	0d 14h 9m 14s	1/4	PING OK - Packet loss = 0%, RTA = 0.13 ms
	S3 bucket usage	OK	09-16-2022 10:10:59	0d 14h 8m 55s	1/4	S3 OK - 7d037bb2-351e-11ed-9012-0242ac110002-data: 0 objects, 0m
proxy	Application frontend	OK	09-16-2022 10:14:06	0d 14h 10m 48s	1/4	SSL OK - data.90.147.174.94.myip.cloud.infn.it - certificate expires in 89 days
	Ping	OK	09-16-2022 10:10:59	0d 14h 8m 55s	1/4	PING OK - Packet loss = 0%, RTA = 0.12 ms
redis	Ping	OK	09-16-2022 10:10:59	0d 14h 8m 55s	1/4	PING OK - Packet loss = 0%, RTA = 0.09 ms
	Redis Service	OK	09-16-2022 10:14:25	0d 14h 10m 29s	1/4	TCP OK - 0.001 second response time on redis port 6379

Results 1 - 16 of 16 Matching Services

https://status.<IP>.myip.cloud.infn.it

References:

User guides:

- Docker compose:
  https://guides.cloud.infn.it/docs/users-guides/en/latest/users_guides/sysadmin/compute/docker_compose.html
- Elasticsearch+Kibana:
  https://guides.cloud.infn.it/docs/users-guides/en/latest/users_guides/sysadmin/compute/elasticsearch_kibana.html
- Sync&Share aaS:
  https://guides.cloud.infn.it/docs/users-guides/en/latest/users_guides/sysadmin/storage/sync_and_share_aas.html

Thank you

for your attention!