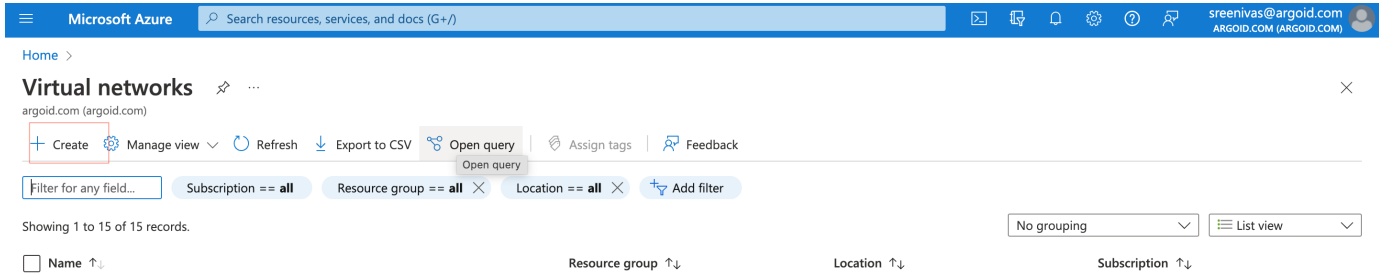


Cluster setup

Create a Resource Group

Step1:- to setup a new cluster we need create a new VPC of that cluster
go to

- Home>Virtual networks>Create



create a new Vpc with the required and at the Address space

- give a new Resource group name and new VPC name and click on next

[Home](#) > [Virtual networks](#) >

Create virtual network

Basics IP Addresses Security Tags Review + create

Azure Virtual Network (VNet) is the fundamental building block for your private network in Azure. VNet enables many types of Azure resources, such as Azure Virtual Machines (VM), to securely communicate with each other, the internet, and on-premises networks. VNet is similar to a traditional network that you'd operate in your own data center, but brings with it additional benefits of Azure's infrastructure such as scale, availability, and isolation. [Learn more about virtual network](#)

Project details

Subscription * ⓘ Microsoft Azure Sponsorship ✓

Resource group * ⓘ (New) test_vasu ✓
[Create new](#)

Instance details

Name * tesd_dev_env ✓

Region * Central India ✓

[Review + create](#)

[< Previous](#)

[Next : IP Addresses >](#)

[Download a template for automation](#)

- Select the required ad
- In the address, space add the new address space
- Add the new subnet range

Show portal menu

networks >

Create virtual network

...

Basics

IP Addresses

Security

Tags

Review + create

The virtual network's address space, specified as one or more address prefixes in CIDR notation (e.g. 192.168.1.0/24).

IPv4 address space

172.19.0.0/16172.19.0.0 - 172.19.255.255 (65536 addresses)

☐ Add IPv6 address space ⓘ

The subnet's address range in CIDR notation (e.g. 192.168.1.0/24). It must be contained by the address space of the virtual network.

+ Add subnet

Remove subnet

<input type="checkbox"/> Subnet name	Subnet address range	NAT gateway
<input type="checkbox"/> default	172.19.0.0/24	-

i

Use of a NAT gateway is recommended for outbound internet access from a subnet. You can deploy a NAT gateway and assign it to a subnet after you create the virtual network. [Learn more](#) ⓘ

Review + create

< Previous

Next : Security >

[Download a template for automation](#)

https://portal.azure.com/#

- In the security keep everything as default

Create virtual network ...

Basics IP Addresses **Security** Tags Review + create

BastionHost ⓘ ☒ Disable
☐ Enable

DDoS Protection Standard ⓘ ☒ Disable
☐ Enable

Firewall ⓘ ☒ Disable
☐ Enable

- Add some tags

Create virtual network ...

Basics IP Addresses Security **Tags** Review + create

Tags are name/value pairs that enable you to categorize resources and view consolidated billing by applying the same tag to multiple resources and resource groups. [Learn more about tags](#) ↗

Note that if you create tags and then change resource settings on other tabs, your tags will be automatically updated.

Name ⓘ	Value ⓘ
<input type="text"/>	: <input type="text"/>

Review + create

< Previous

Next : Review + create >

[Download a template for automation](#)

<https://go.microsoft.com/fwlink/?linkid=873112>

- next review and create a new VPC

Create some VMs as per the newly created resource groups

Microsoft Azure

Search resources, services, and docs (G+)

Home > Virtual machines >

Create a virtual machine ...

Basics

Disks

Networking

Management

Advanced

Tags

Review + create

Create a virtual machine that runs Linux or Windows. Select an image from Azure marketplace or use your own customized image. Complete the Basics tab then Review + create to provision a virtual machine with default parameters or review each tab for full customization. [Learn more](#)

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * ⓘ

Resource group * ⓘ

Microsoft Azure Sponsorship

(New) Resource group

tes

test_vasu

test_vasu1

Instance details

Virtual machine name * ⓘ

Region * ⓘ

Availability options ⓘ

Security type ⓘ

Image * ⓘ

(Asia Pacific) Central India

No infrastructure redundancy required

Standard

CentOS-based 7.6 - Gen2

[See all images](#) | [Configure VM generation](#)

Azure Spot instance ⓘ

Review + create

< Previous

Next : Disks >

(or)

we can create the new VMs using a command but for that we have to create a new security group

1. to create a new security group

[Home](#) > [Network security groups](#) >

Network security g...

argoid.com (argoid.com)

+ Create ⚙️ Manage view ▾ ⋮

Filter for any field...

Name ↑↓

aks-agentpool-27823456-nsg	...
argoid-ansible-1-nsg	...
argoid-ansible-test-1-nsg	...
argoid-ansible-test-11-nsg	...
argoid-demo-hungama-host-004-nsg	...
argoid-demo-hungama-host-005-nsg	...
argoid-demo-hungama-host-006-nsg	...
argoid-demo-hungama-host-007-nsg	...

Create network security group ...

Basics Tags Review + create

Project details

Subscription * Microsoft Azure Sponsorship ▾

Resource group * test_vasu ▾
[Create new](#)

Instance details

Name * test_vasu_nsg ✓

Region * Central India ▾

After creating a network security group we can run the following command to create a new VM

Azure VM Creation using Cli

```
az vm create --name argoid-saas-test-host-004 --resource-group  
test_vasu --location centralindia --image "OpenLogic:CentOS:7_6-gen2:  
latest" --size Standard_B2s --authentication-type ssh --admin-  
username vasu --ssh-key-values ~/.ssh/vasu-dev.pub --storage-sku  
Standard_LRS --os-disk-size-gb 30 --vnet-name argoid-saas1-stage-vpc  
--subnet test_vasu --nsg test_vasu_nsg --private-ip-address 172.19.0.2  
--public-ip-address ""
```

Login to the gateway instance and run the following commands for the setup..

- Install the following services

1. Git

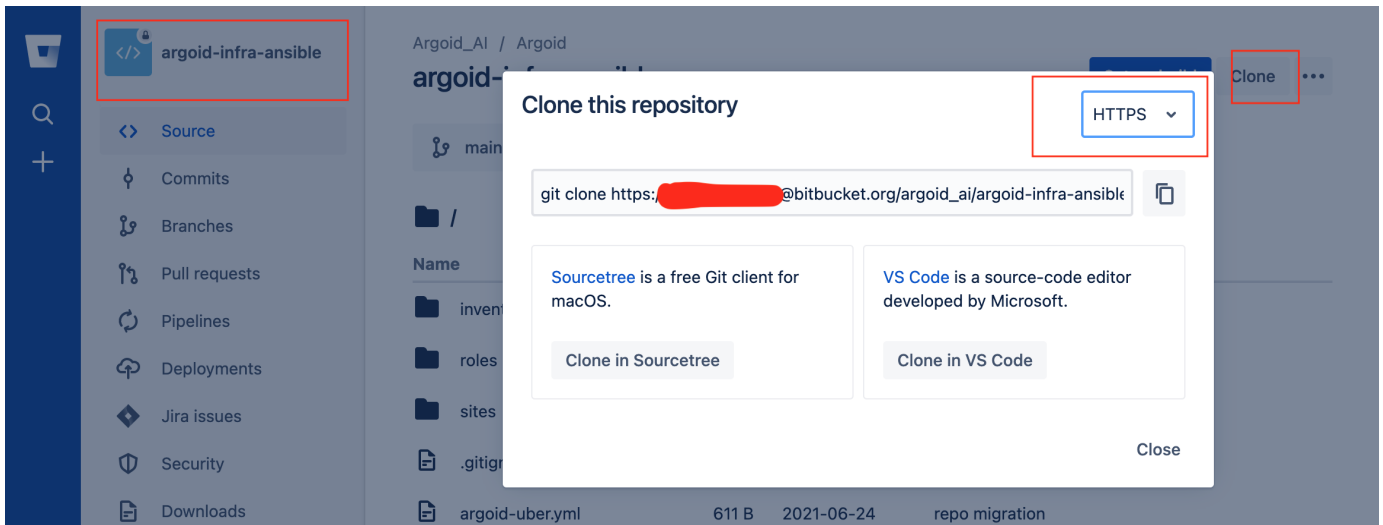
```
yum install git
```

1. Ansible

```
yum install epel-release
```

```
yum install ansible.noarch
```

- Clone argoid infra ansible bit bucket



```
git clone https://<username>@bitbucket.org/argoid_ai/argoid-infra-ansible.git
git checkout dev
git checkout feature/saas-dev-env-installations
```

```
mkdir argoid-infra-ansible/inventory/dev-inventories
```

copy the stage inventory to the new dev inventory

```
cd argoid-infra-amsible
```

```
cp inventory/stage-inventories/stage-common.ini inventory/dev-inventories/dev-common.ini
```

```
vi inventory/dev-inventories/dev-common.ini
```

- Open the inventory file and change the lps of gateway and all the instances

```

[ssh_gateway]
192.1.1.4

[ssh_gateway:vars]
create_ssh_private_keys=True
enable_superuser_privileges=True
ssh_superusers=["chackra", "jeyaraj"]

[docker_parent:children]
docker
airflow
postgres
grafana
prometheus
prometheus_node
alertmanager
nginx
redash

[docker_parent:vars]
docker_edition='ce'
docker_package="docker-{{ docker_edition }}"
docker_package_state=present
docker_yum_repo_url=https://download.docker.com/linux/{{ (ansible_distribution == "Fedora") | ternary("fedora","centos") }}/docker-{{ docker_edition }}.repo
docker_yum_gpg_key=https://download.docker.com/linux/centos/gpg
docker_compose_version="1.23.2"
docker_venv_path=/opt/docker_venv
docker_installation_python_path=/usr/bin/python
docker_python_interpreter={{ docker_venv_path }}/bin/python

[docker]
192.1.1.5
192.1.1.6
192.1.1.7
192.1.1.8
192.1.1.9
192.1.1.10
192.1.1.11
192.1.1.12
192.1.1.13

[airflow]
192.1.1.7

[airflow:vars]
airflow_webserver_port=8082
airflow_enable_statsd_monitoring=True
airflow_dir=/home/airflow/airflow
airflow_user=airflow
airflow_dags_dir=/opt/airflow/dags
airflow_executor=SequentialExecutor
plugins_folder=/home/airflow/airflow/plugins
dag_processor_manager_log_location=/home/airflow/airflow/logs/dag_processor_manager/dag_processor_manager.log
child_process_log_directory=/home/airflow/airflow/logs/scheduler
base_log_folder=/home/airflow/airflow/logs
python3_virtualenv_path=/opt/python3_venv
python_bin_path=/opt/python3_venv/bin/python

```

11

add all the ips in the node_exporter section


```
[postgres:vars]
~ --zsh)_mount_data_dir=/data/1/postgres_data
postgres_data_dir=/var/lib/postgresql/data
postgres_password=postgres
postgres_port=5432
postgres_container_name=postgres-12
postgres_docker_image=postgres:12
postgres_exporter_container_name=postgres_exporter
postgres_exporter_docker_image=wrouesnel/postgres_exporter
postgres_exporter_port=9187
postgres_data_source_name=postgresql://postgres:{{postgres_password}}@{{ansible_default_ipv4.address}}:{{postgres_port}}/?sslmode=disable

#Nifi installed in /opt/nifi/nifi-1.13.2/ directory
[nifi]
192.1.1.6
192.1.1.13

[nifi:vars]
nifi_dir=/opt/nifi
#nifi_source_url=https://archive.apache.org/dist/nifi/1.13.2/nifi-1.13.2-bin.tar.gz
nifi_tar_package=nifi-1.15.1-bin.tar.gz
nifi_source_url="https://archive.apache.org/dist/nifi/1.15.1/{{nifi_tar_package}}"
force=yes
nifi_port=8080
nifi_user=argoid
nifi_jmx_port=9341
nifi_java_heap_size=612

[node_exporter]
192.1.1.5
192.1.1.6
192.1.1.7
192.1.1.8
192.1.1.9
192.1.1.10
192.1.1.11
192.1.1.12
192.1.1.13

[node_exporter:vars]
node_exporter_source_url=https://github.com/prometheus/node_exporter/releases/download/v1.2.2/node_exporter-1.2.2.linux-amd64.tar.gz
node_exporter_dir= /opt/node_exporter
node_exporter_port=9100
force=yes
node_exporter_version=node_exporter-1.2.2.linux-amd64

[kafka]
192.1.1.6

[kafka:vars]
kafka_data_dir=/data/1/kafka
kafka_port=9092
kafka_jmx_port=39321
kafka_heap_size=10
socket_send_buffer_bytes=1024000
socket_receive_buffer_bytes=1024000
socket_request_max_bytes=10248500
log_retention_hours=168
/node_exporter
```

- To create the argoid user

```
##ansible-playbook -i inventory/dev-inventories/dev-common.ini sites
/argoid-user-site.yml --private-key=roles/ansible-ssh-user/files
/id_rsa --limit=192.1.1....
```

- To create leads and ifra members users in all the newly created VMs

```
ansible-playbook -i inventory/dev-inventories/dev-common.ini sites
/argoid-team-users.yml --limit=192.1.1.... --tags=create_ssh_users,
create_ssh_superuser
```

- To install java,bigtop,jmx_exporter in newly created VM's

```
ansible-playbook -i inventory/dev-inventories/dev-common.ini sites
/host-setup.yml --limit=192.1.1....
```

- To install node_exporter in newly created VM's

```
ansible-playbook -i inventory/dev-inventories/dev-common.ini sites
/node_exporter-site.yml --limit=192.1.1...
```

To setup a single node HDFS cluster

Change the IP address to the required node..

```
main_node_manager_java_opts="-Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote -Dcom.sun
jmx_exporter_java_agent_file}=23305:{{jmx_exporter_config_file}} "
```

```
yarn_ha_enabled={{ groups.yarn_resource_manager | count > 1 }}
yarn_timeline_server_web_url={{ hostvars[inventory_hostname].groups['yarn_timeline_server'][0] }}:{{ yarn_timeline_web_po
yarn_log_server_url={{ hostvars[inventory_hostname].groups['yarn_mapreduce_jobhistory_server'][0] }}:{{ jobhistory_web_p
yarn_mapreduce_jobhistory_address={{ hostvars[inventory_hostname].groups['yarn_mapreduce_jobhistory_server'][0] }}:{{ jo
spark_history_server_web_url={{ hostvars[inventory_hostname].groups['spark_history_server'][0] }}:{{ spark_history_serve
```

```
[hdfs:children]
hdfs_namenode
hdfs_datanode
hdfs_journalnode
```

```
[hdfs_namenode]
192.1.1.1
```

```
[hdfs_datanode]
192.1.1.2
```

```
[hdfs_journalnode]
192.1.1.3
```

```
[yarn:children]
yarn_resource_manager
yarn_nodemanager
yarn_timeline_server
yarn_mapreduce_jobhistory_server
spark_history_server
hadoop_clients
```

```
[yarn_resource_manager]
192.1.1.4
```

```
[yarn_nodemanager]
192.1.1.5
```

```
[yarn_timeline_server]
192.1.1.6
```

```
[yarn_mapreduce_jobhistory_server]
192.1.1.7
```

```
[spark_history_server]
192.1.1.8
```

```
[hadoop_clients]
192.1.1.9
192.1.1.10
192.1.1.11
```

```
[hive:children]
```

- For the following variables change the name from stage to the new environments name(dev)

1. hdfs_cluster_name
2. yarn_resource_manager_zk_path
3. yarn_resource_manager_cluster_id
4. yarn_resource_manager_ha_automatic_failover_zk_base_path

```

300 # hdfs_bootstrap=true
306 # hdfs_cluster_name - hdfs-nameservice name
307 hdfs_cluster_name="argoids-dfs"
308 # hdfs_fs_trash_interval - trash retention time in minutes
309 # hdfs_min_block_size=1000
345 yarn_nodemanager_resource_cpu_vcores=4
346 yarn_resourcemanager_zk_path=/yarn-argoidsa-1
347 yarn_resourcemanager_cluster_id=arg-1-yarn
348 yarn_resourcemanager_ha_automatic_failover_zk_base_path=/yarn-saa-1-ha
349 spark_source_url="http://{{ argoid_bigtop_ip }}/argoid_repo/spark/spark-2.3.1-bin-hadoop2.7.tgz"
350 spark_dir=/opt/spark

```

- Ensure that property is set to groups.hdfs_namenode rather than hdfs_namenode

```

325 hdfs_namenodes={{ groups.hdfs_namenode }}
326 hdfs_ha_enabled={{ groups.hdfs_namenode | count > 1 }}
327 hdfs_nameservices={{ hdfs_cluster_name }}
328 hdfs_default_fs="hdfs://{{ hdfs_nameservices if hdfs_ha_enabled else groups.hdfs_namenode[0] + ':8020' }}"
329 jobhistoryserver_enabled={{ groups.history_jobhistoryserver | count > 0 }}
330 jobhistoryserver_address={{ hostvars[inventory_hostname].groups['history_jobhistoryserver'][0] }}:10020
331 jobhistoryserver_webapp_address={{ hostvars[inventory_hostname].groups['history_jobhistoryserver'][0] }}:19888

```

- Check for the Hadoop name node, journal node and data node directory.

```

311 hdfs_dfs_datanode_du_reserved=1073741824
312 hdfs_dfs_journalnode_edits_dir=/data/1/dfs/jn
313 hdfs_namenode_dir=/data/1/dfs/nn
314 hdfs_datanode_dir_list=['/data/1/dfs/data']
315 hadoop_log_maxfilesize="256MB"

```

- First, we need to install a zookeeper

```

###ansible-playbook -i inventory/dev-inventories/dev-common.ini sites
/zookeeper-site.yml --limit=192.1.1... --tags=cluster_setup,start,check

```

- Next to install the Hadoop cluster in a single node.

```

###ansible-playbook -i inventory/dev-inventories/dev-common.ini sites
/hdfs-site.yml --limit=192.1.1... --tags=format_namenode,setup_hdfs

```

- To Install yarn in the next node..

```

###ansible-playbook -i inventory/dev-inventories/dev-common.ini sites
/yarn-site.yml --limit=192.1.1.... --tags=setup_yarn,
setup_historyserver,setup_hadoop_client,setup_spark_client

```

- To install hadoop-clients in the required Vm's

1. In Inventory file under the hadoop-client section give the ip addresses of the VMs in which hadoop-client needs to be installed

```

401
402 [hadoop clients]
403 192.1.1.1
404 192.1.1.2
405 192.1.1.3
406
407 [hive children]

```

```

###ansible-playbook -i inventory/dev-inventories/dev-common.ini sites
/yarn-site.yml --limit=192.1.1... --tags=setup_hadoop_client,
setup_spark_client,setup_yarn

```

- To install docker in the required VM's

1. In Inventory file under the docker section give the ip addresses of the VMs in which docker needs to be installed

```

88 [docker]
89 192.1.1.1
90 192.1.1.2
91 192.1.1.3
92 192.1.1.4
93 192.1.1.5
94 192.1.1.6
95 192.1.1.7
96 192.1.1.8
97 192.1.1.9
98

```

```

###ansible-playbook -i inventory/dev-inventories/dev-common.ini sites
/docker.yml --limit=192.1.1...

```

- To install kafka in the node

1. In Inventory file under the Kafka section give the ip addresses of the VMs in which Kafka needs to be installed

```
484  
485  
486 [kafka]  
487 192.1.1.1  
488  
489 [kafka:vars]  
490 kafka_data_dir=/data/1/kafka
```

```
###ansible-playbook -i inventory/dev-inventories/dev-common.ini sites  
/kafka-site.yml --limit=192.1.1... --tags=cluster_setup,setup_kafka
```

- To install zeppelin in the node

```
[zeppelin]  
192.1.1.1  
[zeppelin:vars]  
zeppelin_source_url=https://arc42.org/zeppelin/releases/0.8.1/zeppelin-0.8.1-bin-  
zeppelin_dir=/opt/zeppelin  
zeppelin_port=8080  
force=yes  
zeppelin_jmx_port=40023  
zeppelin_version=zeppelin-0.8.1  
zeppelin_jvm_heap_size=1024
```

```
####ansible-playbook -i inventory/dev-inventories/dev-common.ini  
sites/zeppelin-site.yml --limit=192.1.1... --tags=cluster_setup,  
setup_zeppelin,check,start
```

- To install nifi in the node

```
#Nifi installed in /opt/nifi/nifi-1.13.2/ directory
[nifi]
192.1
192.1

[nifi:vars]
nifi_dir=/opt/nifi
#nifi_source_url=https://archive.apache.org/dist/nifi/1.13.2/nifi-1.13.2-bin.tar.gz
nifi_tar_package=nifi-1.15.1-bin.tar.gz
nifi_source_url="https://archive.apache.org/dist/nifi/1.15.1/{{nifi_tar_package}}"
force=yes
nifi_port=8080
nifi_user=argoid
nifi_jmx_port=9341
nifi_java_heap_size=612
```

```
###ansible-playbook -i inventory/dev-inventories/dev-common.ini sites
/nifi-site.yml --limit=192.1.1... --tags=setup_nifi,cluster_setup,
check,start
```

- To install airflow

1. In Inventory file under the airflow, section give the ip addresses of the VMs in which airflow needs to be installed and under the python virtual env section

```
98
99 [airflow]
100 192.1...
101
102 [airflow:vars]
103 airflow_webserver_port=8082
104 airflow_enable_statsd_monitoring=True
105 airflow_dir=/home/airflow/airflow
106 airflow_user=airflow
242
243 [python3_venv]
244 192.1...
245
246 [python3_venv:vars]
247 python_venv_version=3.6
248 python3_virtualenv_path=/opt/python3_venv
249
250
```

- and ensure python3_virtualenv_path is same in airflow and python3_venv variables in the inventory

```

112 base_log_folder=/home/airflow/airflow/logs
113 python3_virtualenv_path=/opt/python3_venv
114 python_bin_path=/opt/python3_venv/bin/python
115 python_bin_path_airflow=/opt/python3_venv/bin/airflow
242 ##airflow postgres database user and password creation
243 [python3_venv]
244 192.1.1.
245
246 [python3_venv:vars]
247 python_venv_version=3.6
248 python3_virtualenv_path=/opt/python3_venv
249
250

```

- and ensure that wtforms version is mentioned in the airflow packages section

```

114 python_bin_path=/opt/python3_venv/bin/python
115 python_bin_path_airflow=/opt/python3_venv/bin/airflow
116 ##airflow postgres database user and password creation
117 airflow_packages=[ "SQLAlchemy==1.3.20", "apache-airflow[postgres]==2.0.1", "apache-airflow[statsd]", "apache-airflow[cncf.kubernetes]", "wtforms==2.3.3"]
118 airflow_postgres_db=airflow
119 airflow_postgres_user=airflow
120 airflow_postgres_password=airflow
121 ##statsd_monitoring_paths

```

- To install nginx

```

###ansible-playbook -i inventory/dev-inventories/dev-common.ini sites
/nginx-site.yml --limit=192.1.1.9 --tags=setup_airflow,cluster_setup,
check,start

```

- Haproxy and APISIX are manually installed..
- Haproxy installations
- Follow the below documentation for HAproxy installations..

HAProxy Installation

- APISIX installations
- Follow the below documentation for APISIX installation
- [APISIX and APISIX dashboard installations](#)