



TBB Yetiřtirme Programı Proje Bitirme Ödevi

07.05.2021

Hazırlayan Bilgisi

Adı Soyadı : Büřra KILIÇ
E-Mail Bilgisi : kllcbusra13@gmail.com
Telefon Numarası : +90 (544) 244 45 40



Table of Contents

1 GİRİŞ	4
1.1 Örnek Uygulama Seçilmesi	4
1.2 Git Versiyon Kontrol	4
1.3 Seçilen Örnek uygulamanın Dockerize Edilmesi.....	4
1.4 Kubernetes Yaml's ve Deployment	6
1.5 Haproxy ile Load Balancer	13
1.6 DNS Server Konfigürasyonları.....	14
1.7 JENKINS CI/CD YÖNETİMİ	17
1.8 Sonarqube ile Kod Kalite Kontrolü	22
1.9 Monitoring Süreçleri.....	23
1.10 Ansible ile rsyslog, netsus ve tcpdump kurulumu	28
1.11 OWASP.....	33
1.12 Powershell Scripting	34

TBB Yetiştirme Programı Bitirme Ödevi	12.07.2021
---------------------------------------	------------

192.168.31.101	Kubespray, Haproxy
192.168.31.101:30080	Myflask preprod ortamı
192.168.31.101:30081	Myflask test ortamı
192.168.31.101:30082	Myflask prod ortamı
192.168.31.171	DNS Server
192.168.31.205	Jenkins Sunucusu
192.168.99.108	Minikube
Ana Bilgisayar	Zabbix Agent
192.168.31.130	Zabbix Sunucusu
localhost:3000	Grafana

Hazırlayan	Büşra KILIÇ
------------	-------------

1 GİRİŞ

1.1 Örnek Uygulama Seçilmesi

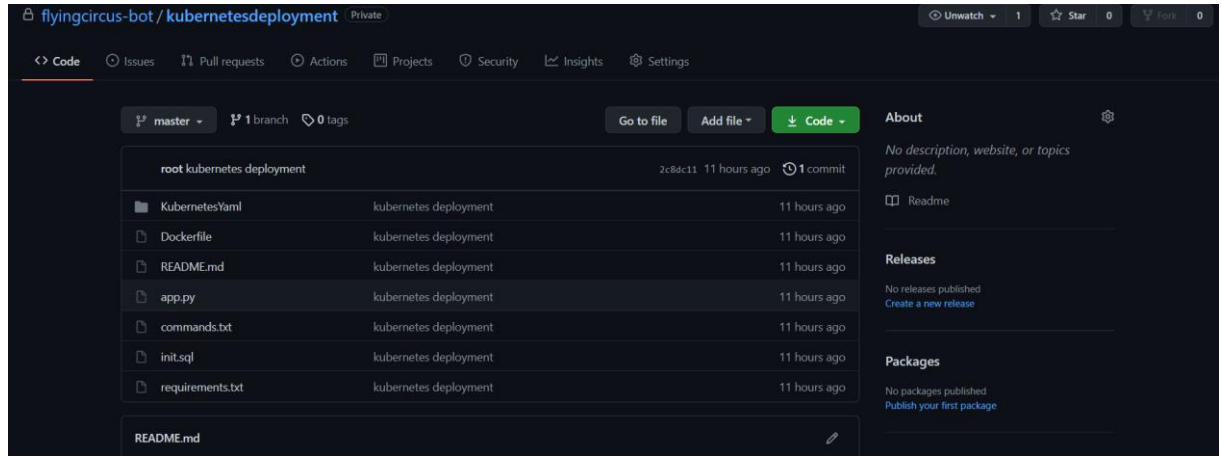
Örnek uygulama olarak aşağıdaki link üzerinde bulunan tbbdevops adlı proje kullanılmıştır.

<https://github.com/aydemirkala/tbbdevops.git>

1.2 Git Versiyon Kontrol

Dockerfile ve KubernetesYaml dosyalarını içerek repository linki aşağıdaki gibidir.

<https://github.com/flyingcircus-bot/kubernetesdeployment>



1.3 Seçilen Örnek uygulamanın Dockerize Edilmesi

Docker network oluşturulur ve build edilir.

```
docker build --network=host -t tbbpython .
docker volume create mysql_volume
docker network create -d bridge mynet
docker run --network=mynet -d -h=mydb --name=mydb -e MYSQL_ROOT_PASSWORD=12345 -p 3306:3306 -v mysql_volume:/var/lib/mysql mysql
docker run --network=mynet -d -h=myapp --name=myapp -p 5001:5001 tbbpython
```

ID	Image	Command	Created	Up	Ports
2de0bef5f4c5	tbbpython	"/bin/sh -c 'python -"	4 days ago	Up 2 minutes	0.0.0.0:5001->5001/tcp, :::5001->5001/tcp
5272259606ef	mysql	"docker-entrypoint.s"	4 days ago	Up 2 minutes	0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp

Dockerfile içeriği aşağıdaki gibidir. Dockerize file içeriğinde uygulama imaj bilgisi, working directory adresi, port bilgisi, requirement yüklenme aşamaları ve tbbdevops uygulamasına entegre edilmesi aşamaları bulunmaktadır. Son olarak ENV aracılığıyla daha sonra deploy edilecek MYSQL authorization bilgileri bir değişken olarak belirtilmiştir.

```
root@master:~/tbbdevops# cat Dockerfile
FROM python:3.6

EXPOSE 5001

WORKDIR /app

COPY requirements.txt /app
RUN pip install -r requirements.txt

COPY app.py /app
CMD python app.py

ENV MYSQL_USERNAME="root"
ENV MYSQL_PASSWORD="12345"
ENV MYSQL_INSTANCE_NAME="myapp"
ENV MYSQL_PORT_3306_TCP_ADDR="mydb"
ENV MYSQL_PORT_3306_TCP_PORT=3306
```

Dockerfile build edilip sonrasında Dockerhub üzerine push edilir.

```
root@master:~/tbbdevops# docker build -t myflask .
Sending build context to Docker daemon 90.62kB
Step 1/12 : FROM python:3.6
--> 5fad76530472
Step 2/12 : EXPOSE 5001
--> Using cache
--> 6a30f67f7cd7
Step 3/12 : WORKDIR /app
--> Using cache
--> e9483f9d838f
Step 4/12 : COPY requirements.txt /app
--> Using cache
--> 867elf85e295
Step 5/12 : RUN pip install -r requirements.txt
--> Using cache
--> dclaa6a8558e
Step 6/12 : COPY app.py /app
--> Using cache
--> b0a08d12ff8f
Step 7/12 : CMD python app.py
--> Using cache
--> 88caa0b7d785
Step 8/12 : ENV MYSQL_USERNAME="root"
--> Using cache
--> c406bfdc7765
Step 9/12 : ENV MYSQL_PASSWORD="12345"
--> Using cache
--> db08adf10481
Step 10/12 : ENV MYSQL_INSTANCE_NAME="myapp"
--> Using cache
--> 2949aa413668
Step 11/12 : ENV MYSQL_PORT_3306_TCP_ADDR="mydb"
--> Using cache
--> eaa939e53ca6
Step 12/12 : ENV MYSQL_PORT_3306_TCP_PORT=3306
--> Using cache
--> 6ec16ddb59d9
Successfully built 6ec16ddb59d9
Successfully tagged myflask:latest
```

```
root@master:~/tbbdevops# docker tag myflask flyingcircusbot/myflask
```

```
root@master:~/tbbdevops# docker push flyingcircusbot/myflask:latest
```

flyingcircusbot / myflask

This repository does not have a description

Last pushed: 4 hours ago

Docker commands

To push a new tag to this repository,

```
docker push flyingcircusbot/myflask:tagname
```

Tags and Scans

This repository contains 5 tag(s).

VULNERABILITY SCANNING - DISABLED [Enable](#)

TAG	OS	PULLED	PUSHED
v4	linux/amd64	8 minutes ago	4 hours ago
v3	linux/amd64	8 minutes ago	7 hours ago
v2	linux/amd64	8 minutes ago	7 hours ago
v1	linux/amd64	8 minutes ago	7 hours ago

Automated Builds

Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available on Pro and Team plans.

[Upgrade to Pro](#) [Learn more](#)

Sonrasında docker imaj run edilir ve 5001 portu üzerinden API çıktıları incelenir.

```
root@master:~/tbbdevops# docker run -d flyingcircusbot/myflask
b95a9bd6074abd74eb07602c27dfb111d4cb533119ff0d7329039e22c5d9d65d
```

```
Not secure | 192.168.31.101:5001
```

```
{ "buyuk_sehirler": [ [ "Istanbul", "34" ], [ "Ankara", "06" ] ] }
```

1.4 Kubernetes Yaml's ve Deployment

Test, prod, preprod adimli üç ayrı namespace oluşturulur.

```
root@master:~/kubernetesdeployment/KubernetesYaml/prod# kubectl get namespaces
```

NAME	STATUS	AGE
default	Active	2d20h
kube-node-lease	Active	2d20h
kube-public	Active	2d20h
kube-system	Active	2d20h
preprod	Active	4s
prod	Active	2d4h
test	Active	2d4h

Mysql-pv.yaml dosyası ile deployment için gerekli olan fiziksel volume create işlemi gerçekleştirilir.

```
--  
apiVersion: v1  
kind: PersistentVolume  
metadata:  
  name: mydb-pv-volume-test  
  namespace: test  
spec:  
  capacity:  
    storage: 0.5Gi  
  accessModes:  
    - ReadWriteOnce  
  hostPath:  
    path: "/opt/mydb-test"
```

Mysql-pvc.yaml dosyası aracılığıyla oluşturulan fiziksel volume'un claim aşaması gerçekleştirilir.

```
--  
apiVersion: v1  
kind: PersistentVolumeClaim  
metadata:  
  name: mydb-pvc  
  namespace: test  
spec:  
  accessModes:  
    - ReadWriteOnce  
  resources:  
    requests:  
      storage: 0.1Gi  
~
```

Mysql-secret.yaml dosyası ile Dockerize ettiğimiz image içeriğinde gerekli olan ENV bilgisi MYSQL_ROOT_PASSWORD hashlenerek implement edilir.

```
--  
apiVersion: v1  
data:  
  MYSQL_ROOT_PASSWORD: MTIzNDU=  
kind: Secret  
metadata:  
  name: mysql-root-password  
  namespace: test  
type: Opaque  
~
```

Mysql-initdb-config.yaml ile init.sql dosyası tanımlanır.

```
--
apiVersion: v1
kind: ConfigMap
metadata:
  name: mysql-initdb-config
  namespace: test
data:
  initdb.sql: |
    CREATE USER 'root'@'%' IDENTIFIED BY 'root';
    GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' WITH GRANT OPTION;
    FLUSH PRIVILEGES;
    CREATE DATABASE IF NOT EXISTS sehirler;
    use sehirler;
    CREATE TABLE buyuk_sehirler (
      isim VARCHAR(20),
      plaka VARCHAR(10)
    );
    INSERT INTO buyuk_sehirler
      (isim, plaka )
    VALUES
      ('Istanbul', '34'),
      ('Ankara', '06');
```

Mysql-service.yaml ile mysql'in üzerinde çalışacağı NodePort belirlenir.

```
--
apiVersion: v1
kind: Service
metadata:
  name: mydb
  namespace: test
  labels:
    app: mydb
spec:
  ports:
    - port: 3306
      protocol: TCP
      targetPort: 3306
      nodePort: 30338
  selector:
    app: mydb
  type: NodePort
```


Mysql-deployment.yaml ile ENV bilgileri de girilerek deploy yaml oluşturulur.

```
--
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: mydb
  namespace: test
  labels:
    app: mydb
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mydb
  serviceName: "mydb"
  template:
    metadata:
      name: mydb
      labels:
        app: mydb
    spec:
      imagePullSecrets:
        - name: secret-key
      containers:
        - name: mysql-test
          image: mysql
          imagePullPolicy: Always
          env:
            - name: MYSQL_ROOT_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: mysql-root-password
                  key: MYSQL_ROOT_PASSWORD
          args: ["--default-authentication-plugin=mysql_native_password"]
          ports:
            - containerPort: 3306
          resources:
            requests:
              cpu: 200m
              memory: 0.5Gi
          livenessProbe:
            exec:
              command: ["mysqladmin", "ping"]
            initialDelaySeconds: 30
            periodSeconds: 10
            timeoutSeconds: 5
          volumeMounts:
            - name: mydb-pv-volume-test
```

Python-service.yaml ile Python uygulamasının çalışacağı namespace ve NodePort belirtilir.

```
--  
apiVersion: v1  
kind: Service  
metadata:  
  name: flask-service  
  namespace: test  
spec:  
  ports:  
    - port: 5001  
      protocol: TCP  
      targetPort: 5001  
      nodePort: 30082  
  selector:  
    app: myapp  
  type: NodePort  
~
```

Son olarak Python_deployment.yaml ile deployment için gerekli bilgiler set edilir.

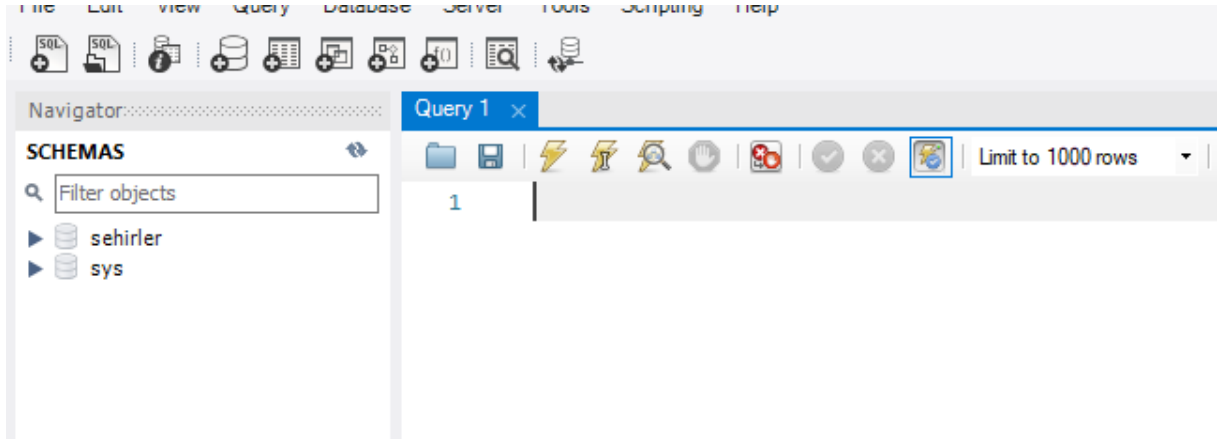
```
--
apiVersion: apps/v1
kind: Deployment
metadata:
  name: myapp-deploy
  namespace: test
  labels:
    app: myapp
spec:
  replicas: 1
  selector:
    matchLabels:
      app: myapp
  template:
    metadata:
      labels:
        app: myapp
    spec:
      imagePullSecrets:
        - name: secret-key
      containers:
        - name: myapp-flask-test
          image: flyingcircusbot/myflask:latest
          imagePullPolicy: Always
          ports:
            - containerPort: 5001
          resources:
            requests:
              cpu: 200m
              memory: 0.5Gi
          livenessProbe:
            httpGet:
              path: /
              port: 5001
            initialDelaySeconds: 30
            periodSeconds: 10
            timeoutSeconds: 5
          env:
            - name: MYSQL_USERNAME
              value: "root"
            - name: MYSQL_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: mysql-root-password
                  key: MYSQL_ROOT_PASSWORD
            - name: MYSQL_INSTANCE_NAME
              value: "myapp"
```

Bahsedilen yaml dosyaları test ve preprod namespacesi için oluşturulur ve sonrasında apply işlemi gerçekleştirilir.

```
root@master:~/kubernetesdeployment/KubernetesYaml/preprod# kubectl apply -f mysql-pv.yaml
persistentvolume/mydb-pv-volume-preprod created
root@master:~/kubernetesdeployment/KubernetesYaml/preprod# kubectl apply -f mysql-pvc.yaml
persistentvolumeclaim/mydb-pvc created
root@master:~/kubernetesdeployment/KubernetesYaml/preprod# kubectl apply -f mysql-service.yaml
service/mydb created
root@master:~/kubernetesdeployment/KubernetesYaml/preprod# kubectl apply -f mysql-initdb-config.yaml
configmap/mysql-initdb-config created
root@master:~/kubernetesdeployment/KubernetesYaml/preprod# kubectl apply -f mysql-secret.yaml
secret/mysql-root-password created
root@master:~/kubernetesdeployment/KubernetesYaml/preprod# kubectl apply -f mysql-deployment.yaml
deployment.apps/mydb created
root@master:~/kubernetesdeployment/KubernetesYaml/preprod# kubectl apply -f python-service.yaml
service/flask-service created
root@master:~/kubernetesdeployment/KubernetesYaml/preprod# kubectl apply -f python-deployment.yaml
deployment.apps/myapp-deploy created

root@master:~/kubernetesdeployment/KubernetesYaml/preprod# kubectl get pods --namespace=preprod
NAME                                READY   STATUS    RESTARTS   AGE
myapp-deploy-86c446fc6-6plsc        1/1     Running   0           73s
mydb-0                               1/1     Running   0           95s
```

Workbench üzerinden kontrolü yapılır.

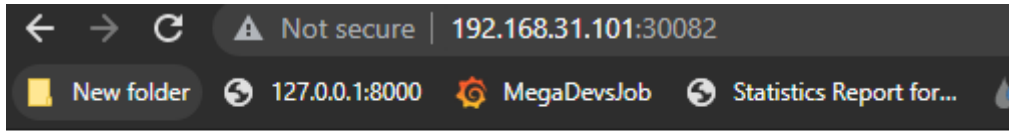


Bu aşamalar test ve prod kısımları için de gerçekleştirilir.

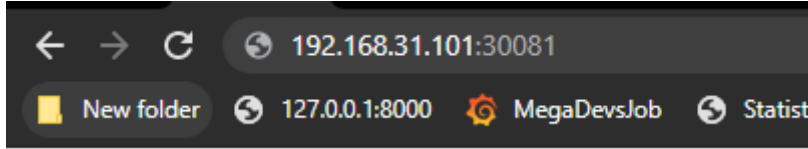
```
preprod    myapp-deploy-86c446fc6-6plsc    1/1    Running   4    52m
preprod    mydb-0                          1/1    Running   6    52m
prod       myapp-deploy-78868d7575-wdw9m   1/1    Running   3    17m
prod       mydb-0                          1/1    Running   5    18m
test       myapp-deploy-57c4bcc68-ns8dw     1/1    Running   9    21m
test       mydb-0                          0/1    Pending   0    28m
```

```
preprod    myapp-deploy-86c446fc6-6plsc    1/1    Running   4    52m
preprod    mydb-0                          1/1    Running   6    52m
prod       myapp-deploy-78868d7575-wdw9m   1/1    Running   3    17m
prod       mydb-0                          1/1    Running   5    18m
test       myapp-deploy-57c4bcc68-ns8dw     1/1    Running   9    21m
```

```
← → ↺ ⚠ Not secure | 192.168.31.101:30080
New folder 127.0.0.1:8000 MegaDevsJob Statistics Report for... On Parmak Q Klavy... CVonline: Ima
{"buyuk_sehirler": [{"Istanbul", "34"}, {"Ankara", "06"}]}
```

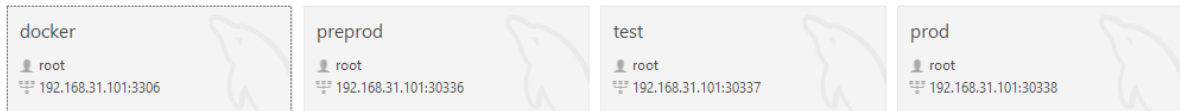


```
{"buyuk_sehirler": [{"Istanbul", "34"}, {"Ankara", "06"}]}
```



```
{"buyuk_sehirler": [{"Istanbul", "34"}, {"Ankara", "06"}]}
```

Workbench üzerinden kontrolü gerçekleştirilir.



1.5 Haproxy ile Load Balancer

Haproxy kurulumu yapılır ve haproxt etc dosyası düzenlenir.

```
root@master:~# systemctl status haproxy.service
● haproxy.service - HAProxy Load Balancer
   Loaded: loaded (/lib/systemd/system/haproxy.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2021-07-10 04:18:35 UTC; 3min 57s ago
     Docs: man:haproxy(1)
           file:/usr/share/doc/haproxy/configuration.txt.gz
   Process: 1198 ExecStartPre=/usr/sbin/haproxy -f $CONFIG -c -q $EXTRA_OPTS (code=exited, status=0/SUCCESS)
  Main PID: 1304 (haproxy)
    Tasks: 2 (limit: 4915)
   CGroup: /system.slice/haproxy.service
           └─1304 /usr/sbin/haproxy -Ws -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid
           └─1306 /usr/sbin/haproxy -Ws -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid
```

Config dosyası içinde load balance IP adresleri ilgili namespaces'lere göre girilir.

```

backend k8s_apiservers
mode tcp
balance leastconn
option ssl-hello-chk
option log-health-checks
default-server inter 10s fall 2
server cluster1-master-1 192.168.31.101:6443 check

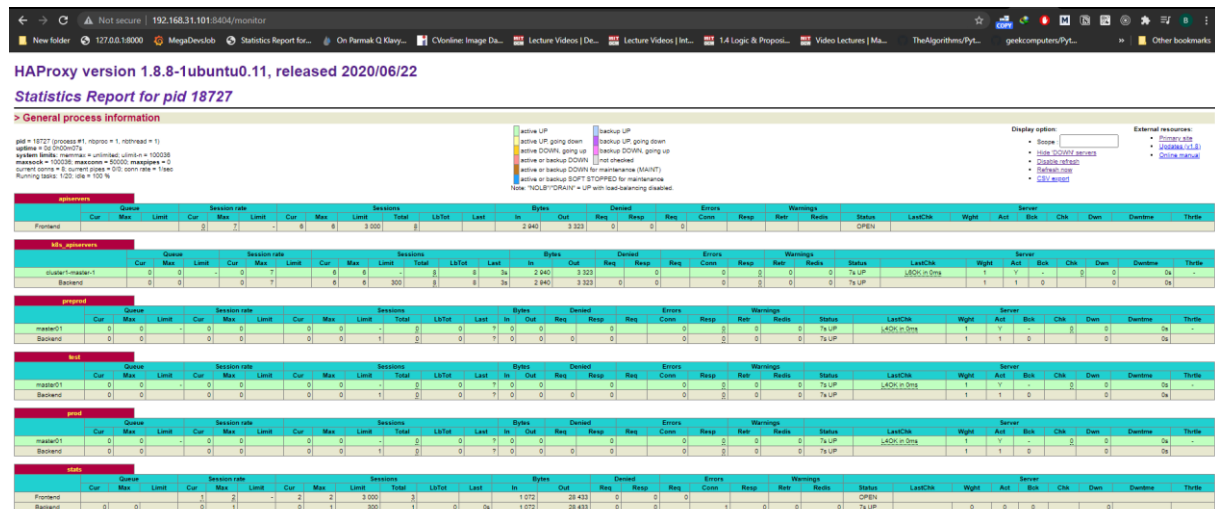
backend preprod
balance roundrobin
mode http
server master01 192.168.31.101:30080 check

backend test
balance roundrobin
mode http
server master01 192.168.31.101:30081 check

backend prod
balance roundrobin
mode http
server master01 192.168.31.101:30082 check

```

Haproxy üzerinde incelenir.



1.6 DNS Server Konfigürasyonları

DNS server'ın bulunacağı makine static IP olarak düzenlenir. DNS server kurulum sonrasında 192.168.31.101 (kubespriy,haproxy) makinesi ile aralarında bağlantı kurulur. Son olarak DNS kayıtları girilir ve haproxy üzerinde konfigürasyon yapılır.

Internet Protocol Version 4 (TCP/IPv4) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

☐ Obtain an IP address automatically

☒ Use the following IP address:

IP address: 192 . 168 . 31 . 171

Subnet mask: 255 . 255 . 255 . 0

Default gateway: 192 . 168 . 31 . 1

☐ Obtain DNS server address automatically

☒ Use the following DNS server addresses:

Preferred DNS server: 192 . 168 . 31 . 171

Alternate DNS server: . . .

☐ Validate settings upon exit

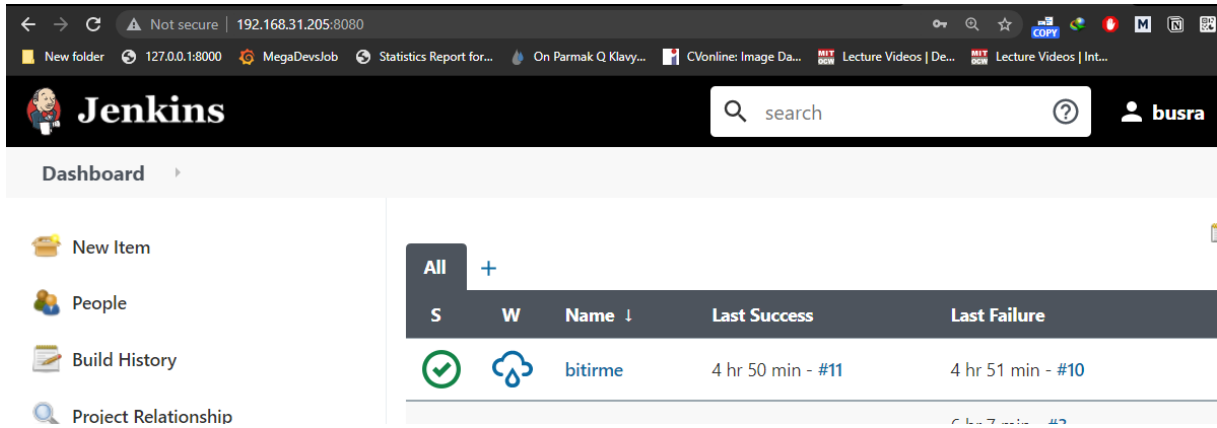
Advanced...

OK Cancel

DNS	Name	Type	Data
WIN-C6H9EFFSDNB	(same as parent folder)	Start of Authority (SOA)	[10], win-c6h9effsdbn, ho...
Forward Lookup Zones	(same as parent folder)	Name Server (NS)	win-c6h9effsdbn.
tbbdemo.com	myflask-tbb-dev	Host (A)	192.168.31.101
tbbprod.com	myflask-tbb-test	Host (A)	192.168.31.101
Reverse Lookup Zones	haproxy	Host (A)	192.168.31.101
Trust Points			
Conditional Forwarders			

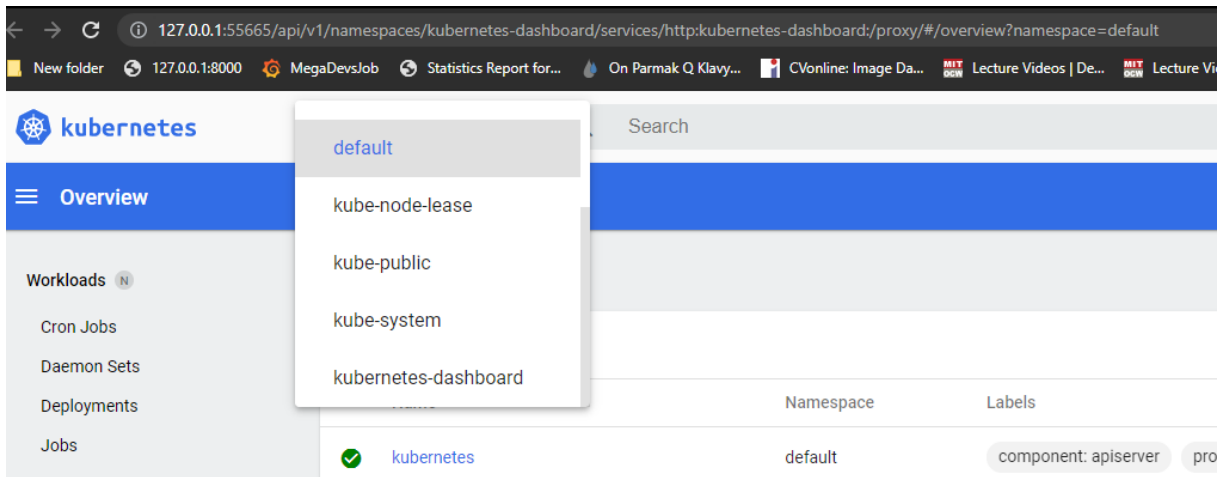
```
# This is the network config written by 'subiquity'
network:
  ethernets:
    enp0s3:
      dhcp4: no
      addresses: [192.168.31.101/24]
      gateway4: 192.168.31.1
      nameservers:
        addresses: [192.168.31.171]
    enp0s8:
      dhcp4: yes
```

```
frontend http
  bind *:80
  mode http
  acl acldev hdr(host) -i myflask-tbb-dev.tbbdemo.com
  use_backend preprod if acldev
  acl acltest hdr(host) -i myflask-tbb-test.tbbdemo.com
  use_backend test if acltest
  acl aclprod hdr(host) -i myflask-tbb-dev.tbbdemo.com
  use_backend prod if aclprod
  acl aclhap hdr(host) -i myflask-tbb-dev.tbbdemo.com
  use_backend haproxy if aclhap
backend preprod
  balance roundrobin
  mode http
  server master01 192.168.31.101:30080 check
backend test
  balance roundrobin
  mode http
  server master01 192.168.31.101:30081 check
backend prod
  balance roundrobin
  mode http
  server master01 192.168.31.101:30082 check
backend haproxy
  balance roundrobin
  mode http
  server master01 192.168.31.101:8404 check
```

Jenkins'in kubernetes tarafı için minikube kuruldu ve Jenkins /var/lib/jenkins/config dosyasında bağlantı kuruldu.

Jenkins üzerindeki deploy aşamalarını daha iyi gözlemleyebilmek amacıyla minikube tarafından kubernetes dashboard kullanıldı.



Kubernetes üzerinde dev, test ve prod adlarında üç ayrı namespace oluşturuldu.

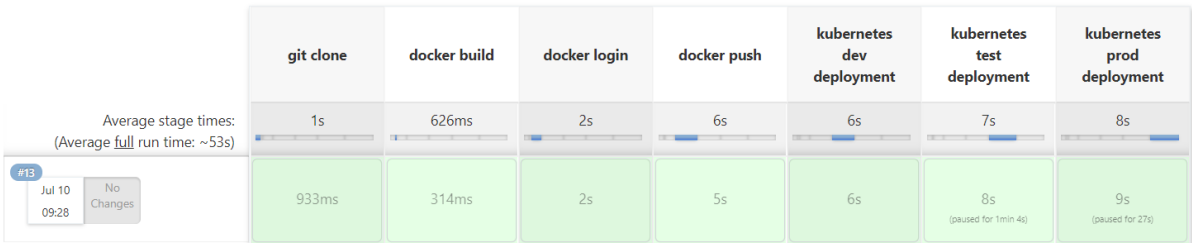
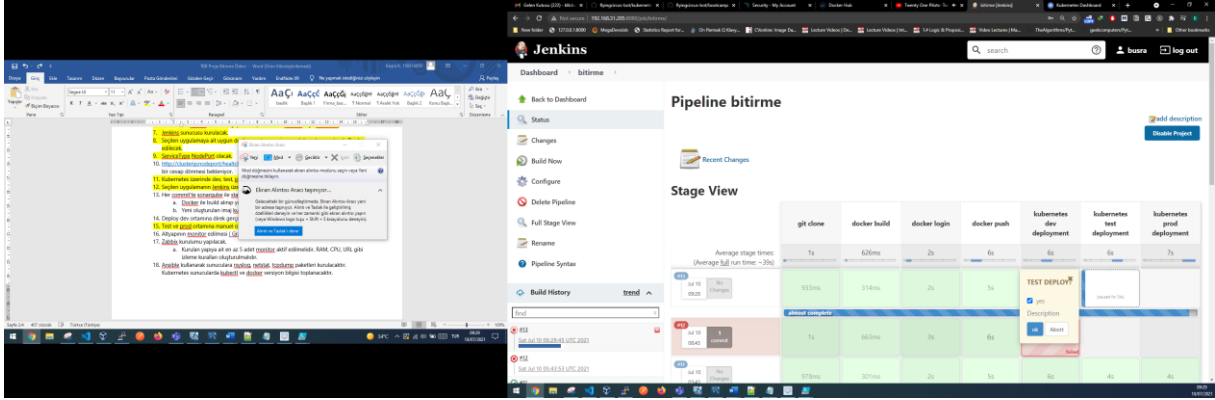
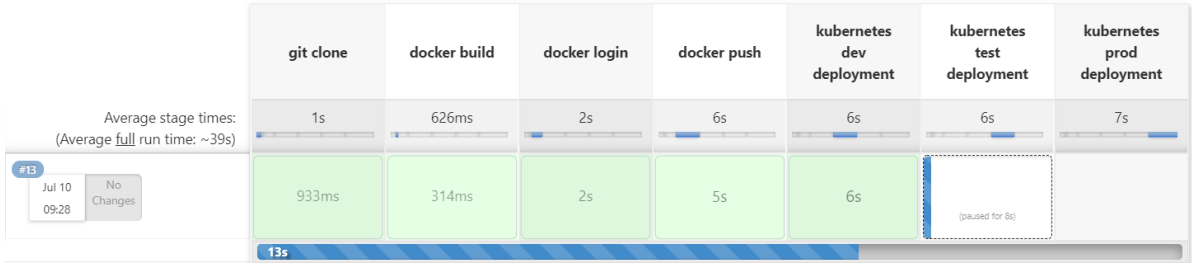
```
jenkins@busra:/home/busra$ kubectl create namespace dev
namespace/dev created
jenkins@busra:/home/busra$ kubectl create namespace test
namespace/test created
jenkins@busra:/home/busra$ kubectl create namespace prod
namespace/prod created
jenkins@busra:/home/busra$ kubectl get pods --all-namespaces
NAMESPACE NAME READY STATUS RESTARTS AGE
kube-system coredns-74ff55c5b-nqtbh 1/1 Running 2 74m
kube-system etcd-minikube 1/1 Running 2 74m
kube-system kube-apiserver-minikube 1/1 Running 2 74m
kube-system kube-controller-manager-minikube 1/1 Running 2 74m
kube-system kube-proxy-cq76h 1/1 Running 2 74m
kube-system kube-scheduler-minikube 1/1 Running 2 74m
kube-system storage-provisioner 1/1 Running 4 74m
kubernetes-dashboard dashboard-metrics-scraper-f6647bd8c-r9hlw 1/1 Running 2 74m
kubernetes-dashboard kubernetes-dashboard-968bcb79-wcx6j 1/1 Running 4 74m
```

Namespaces				
Name ↑	Labels	Phase	Created ↑	
✓ prod	-	Active	59.seconds.ago	⋮
✓ test	-	Active	a.minute.ago	⋮
✓ dev	-	Active	a.minute.ago	⋮
✓ kubernetes-dashboard	addonmanager.kubernetes.io/mode: Reconcile kubernetes.io/minikube-addons: dashboard	Active	an.hour.ago	⋮
✓ default	-	Active	an.hour.ago	⋮
✓ kube-node-lease	-	Active	an.hour.ago	⋮
✓ kube-public	-	Active	an.hour.ago	⋮
✓ kube-system	-	Active	an.hour.ago	⋮
1 - 8 of 8				< < > >

Sunucu üzerinde CI/CD aşamaları gerçekleştirildi. Docker Hub ve Git credential oluşturuldu. Pipeline üzerinde build trigger kullanıldı. Dev ortamını otomatik olarak geçip, test ve prod ortamlarında yaklaşık 1 dakikalık bekleme süresiyle manuel onay beklendi.

```
node {
  stage ("git clone"){
    git credentialsId: 'GITHUB', url: 'https://github.com/flyingcircus-bot/kubernetesdeployment.git'
  }
  stage ("docker build"){
    sh 'docker build -t flyingcircusbot/myflask:v${BUILD_NUMBER} .'
  }
  stage ("docker login"){
    withCredentials([{$class: 'UsernamePasswordMultiBinding', credentialsId: 'DOCKERHUB', usernameVariable: 'USERNAME', passwordVariable: 'PASSWORD'}]) {
      sh 'docker login -u $USERNAME -p $PASSWORD'
    }
  }
  stage ("docker push"){
    sh 'docker push flyingcircusbot/myflask:v${BUILD_NUMBER}'
  }
  stage ("kubernetes dev deployment"){
    script {
      env.DOCKER_BUILD_NUMBER="${BUILD_NUMBER}"
    }
    sh 'echo ${DOCKER_BUILD_NUMBER}'
    sh 'envsubst < ./KubernetesYaml/dev/mysql-deployment.yaml | kubectl apply -f -'
    sh 'envsubst < ./KubernetesYaml/dev/python-deployment.yaml | kubectl apply -f -'
    sh 'kubectl apply -f ./KubernetesYaml/dev/mysql-pv.yaml'
    sh 'kubectl apply -f ./KubernetesYaml/dev/mysql-pvc.yaml'
    sh 'kubectl apply -f ./KubernetesYaml/dev/mysql-secret.yaml'
    sh 'kubectl apply -f ./KubernetesYaml/dev/mysql-initdb-config.yaml'
    sh 'kubectl apply -f ./KubernetesYaml/dev/mysql-service.yaml'
    sh 'kubectl apply -f ./KubernetesYaml/dev/python-service.yaml'
  }
}

stage ("kubernetes test deployment"){
  def deployment= input(message: 'TEST DEPLOY?', ok: 'ok', parameters:[booleanParam(defaultValue:true,description:'Description')])
  if (deployment == true){
    script {
      env.DOCKER_BUILD_NUMBER="${BUILD_NUMBER}"
    }
    sh 'echo ${DOCKER_BUILD_NUMBER}'
    sh 'envsubst < ./KubernetesYaml/test/mysql-deployment.yaml | kubectl apply -f -'
    sh 'envsubst < ./KubernetesYaml/test/python-deployment.yaml | kubectl apply -f -'
    sh 'kubectl apply -f ./KubernetesYaml/test/mysql-pv.yaml'
    sh 'kubectl apply -f ./KubernetesYaml/test/mysql-pvc.yaml'
    sh 'kubectl apply -f ./KubernetesYaml/test/mysql-secret.yaml'
    sh 'kubectl apply -f ./KubernetesYaml/test/mysql-initdb-config.yaml'
    sh 'kubectl apply -f ./KubernetesYaml/test/mysql-service.yaml'
    sh 'kubectl apply -f ./KubernetesYaml/test/python-service.yaml'
  } else {
    echo "Deploy Skipped"
  }
}
```



Dashboard ve jenkins üzerinde kontrolü gerçekleştirildi.

dev

Q

Search

+

Pods

Name	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created <div>↑</div>
<div><div></div>myapp-deploy-68fc6694c5-2jctm</div>	<div>app: myapp</div> <div>pod-template-hash: 68fc6694c5</div>	minikube	Running	1	-	-	5 minutes ago <div></div>
<div><div></div>mydb-0</div>	<div>app: mydb</div> <div>controller-revision-hash: mydb-57d8bc6f9</div> <div>statefulset.kubernetes.io/pod-name: mydb-0</div>	minikube	Running	0	-	-	5 minutes ago <div></div>

1 - 2 of 2

|<

<

>

|>

prod Q Search

Pods

Name	Labels	Node	Status	Restarts	CPU Usage
✓ myapp-deploy-78868d7575-pz7x4	app: myapp pod-template-hash: 78868d7575	minikube	Running	0	-
✓ mydb-0	app: mydb controller-revision-hash: mydb-684ddf949 statefulset.kubernetes.io/pod-name: mydb-0	minikube	Running	0	-

Pods

Name	Labels	Node
✓ myapp-deploy-57c4bcc68-fcwkj	app: myapp pod-template-hash: 57c4bcc68	minikube
✓ mydb-0	app: mydb controller-revision-hash: mydb-857f7dc5fd statefulset.kubernetes.io/pod-name: mydb-0	minikube

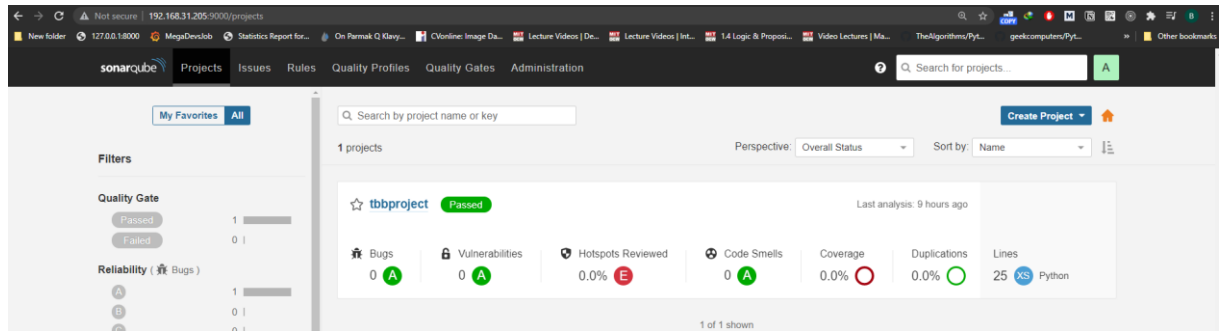
```
jenkins@busra:/home/busra$ kubectl get pods --all-namespaces
```

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
dev	myapp-deploy-68fc6694c5-2jctm	1/1	Running	1	5m33s
dev	mydb-0	1/1	Running	0	5m34s
kube-system	coredns-74ff55c5b-ngtbh	1/1	Running	2	84m
kube-system	etcd-minikube	1/1	Running	2	84m
kube-system	kube-apiserver-minikube	1/1	Running	2	84m
kube-system	kube-controller-manager-minikube	1/1	Running	2	84m
kube-system	kube-proxy-cq76h	1/1	Running	2	84m
kube-system	kube-scheduler-minikube	1/1	Running	2	84m
kube-system	storage-provisioner	1/1	Running	4	84m
kubernetes-dashboard	dashboard-metrics-scraper-f6647bd8c-r9hlw	1/1	Running	2	84m
kubernetes-dashboard	kubernetes-dashboard-968bcb79-wcx6j	1/1	Running	4	84m
prod	myapp-deploy-78868d7575-pz7x4	1/1	Running	0	3m47s
prod	mydb-0	1/1	Running	0	3m47s
test	myapp-deploy-57c4bcc68-fcwkj	1/1	Running	0	4m21s
test	mydb-0	1/1	Running	0	4m23s

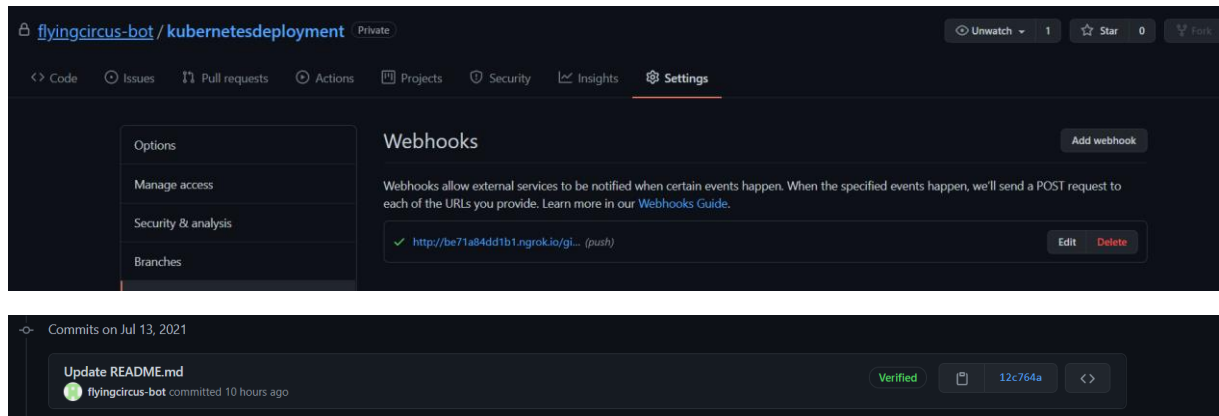
1.8 Sonarqube ile Kod Kalite Kontrolü

Freestyle bir proje oluşturulup sonarqube üzerinde kalite kontrol yapılabilmesi için gerekli ayarlar yapıldı ve aynı şekilde ön ihtiyaçlar karşılandı.

Output çıktısı incelendi.



Commit üzerinde etkilerini görebilmek amacıyla test edildi.



☆ **tbbproject**
Passed

Last analysis: 9 hours ago

🐛 Bugs
0 A

🔒 Vulnerabilities
0 A

🔍 Hotspots Reviewed
0.0% E

💩 Code Smells
0 A

📶 Coverage
0.0%

📄 Duplications
0.0%

📄 Lines
25 XS Python

1.9 Monitoring Süreçleri

Monitoring aşamasında Zabbix ve Grafana kuruldu. Monitoring süreçlerinde RAM, CPU, URL Monitoring gerçekleştirildi.

Zabbix üzerinden Configuration>Hosts sekmesinden monitör edilmesi planlanan makine create edilir. Latest Data sekmesinden son datalar arasında inceleme yapılır.

Zabbix Agent Monitoring – Ana Makine

Host groups: DevOps Select

Tags: And/Or Or

Hosts: 192.168.31.98 Select

Name:

Contains Remove

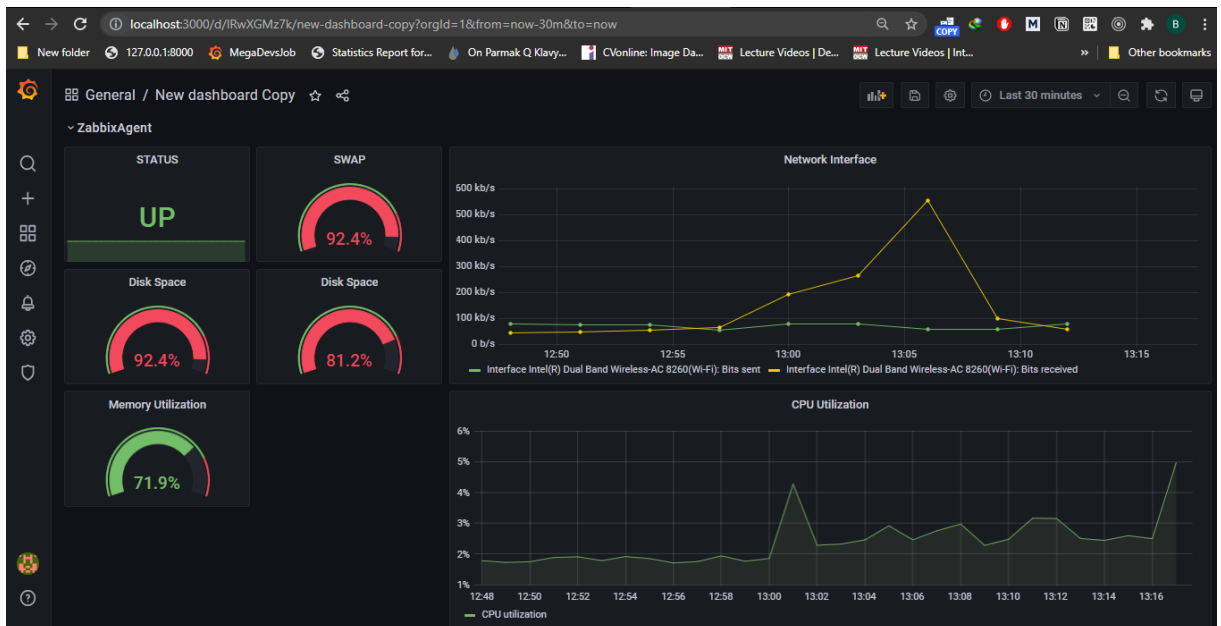
Add

☐ Show details

☒ Show items without data

Apply
Reset

<input type="checkbox"/>	Host	Name	Last check	Last value	Change	Tags	
<input type="checkbox"/>	192.168.31.98	0: Average disk read queue length	2021-07-10 13:19:26	0		Application: Disk 0	Graph
<input type="checkbox"/>	192.168.31.98	0: Average disk write queue length	2021-07-10 13:19:28	0		Application: Disk 0	Graph
<input type="checkbox"/>	192.168.31.98	0: Disk average queue size (avgqu-sz)	2021-07-10 13:18:34	0		Application: Disk 0	Graph
<input type="checkbox"/>	192.168.31.98	0: Disk read rate	2021-07-10 13:18:36	0 r/s		Application: Disk 0	Graph
<input type="checkbox"/>	192.168.31.98	0: Disk read request avg waiting time	2021-07-10 13:19:30	0		Application: Disk 0	Graph
<input type="checkbox"/>	192.168.31.98	0: Disk utilization	2021-07-10 13:19:24	0 %		Application: Disk 0	Graph
<input type="checkbox"/>	192.168.31.98	0: Disk write rate	2021-07-10 13:18:38	0 w/s		Application: Disk 0	Graph
<input type="checkbox"/>	192.168.31.98	0: Disk write request avg waiting time	2021-07-10 13:18:32	0		Application: Disk 0	Graph



Uygulama halihazırda http üzerinden koşulduğu için URL monitoring gerçekleştirildi. Preprod IP, host olarak Zabbix üzerine eklendi ve sonrasında Web Senaryosu hazırlandı.

All hosts / preprod Enabled ZBX Items 2 Triggers 2 Graphs Discovery rules Web scenarios 1

Host Templates 2 IPMI Tags Macros Inventory Encryption Value mapping

* Host name preprod

Visible name preprod

* Groups Kube X type here to search Select

Interfaces	Type	IP address	DNS name	Connect to	Port	Default
Agent		192.168.31.101		IP DNS	30080	<input checked="" type="radio"/> Remove

Add

Description

Monitored by proxy (no proxy) v

Enabled ☒

Update Clone Full clone Delete Cancel

Host Templates 2 IPMI Tags Macros Inventory Encryption Value mapping

Linked templates

Name	Action
HTTP Service	Unlink Unlink and clear
HTTPS Service	Unlink Unlink and clear

Link new templates

type here to search Select

Update Clone Full clone Delete Cancel

All hosts / preprod Enabled ZBX Items 2 Triggers 2 Graphs Discovery rules Web scenarios 1

Scenario Steps 2 Tags Authentication

* Name

* Update interval

* Attempts

Agent

HTTP proxy

Variables

Name	Value	
<input type="text" value="name"/>	<input type="text" value="value"/>	Remove

[Add](#)

Headers

Name	Value	
<input type="text" value="name"/>	<input type="text" value="value"/>	Remove

[Add](#)

Enabled ☒

[Update](#) [Clone](#) [Clear history and trends](#) [Delete](#) [Cancel](#)

Latest Datalar elde edildikten sonrasında Grafana üzerinden panel oluşturuldu. Bu aşamalar Prod ve Test ortamlarına da uygulandı.

Scenario Steps 2 Tags Authentication

* Steps

Name	Timeout	URL	Required	Status codes	Action
1: preprod	15s	http://192.168.31.101:30080/		200	Remove

Hosts [Select](#)

Name

Tags [Remove](#)

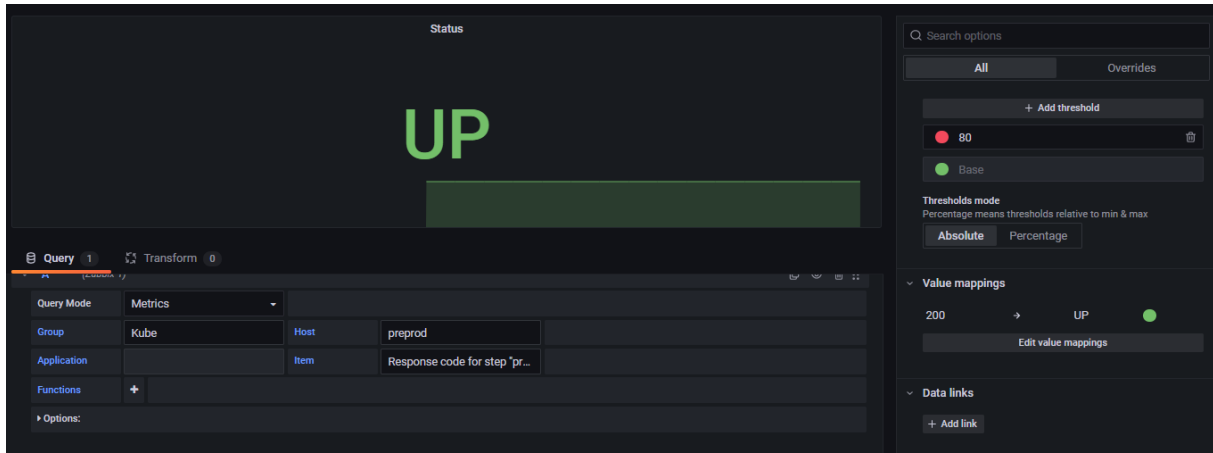
[Add](#)

[Apply](#) [Reset](#)

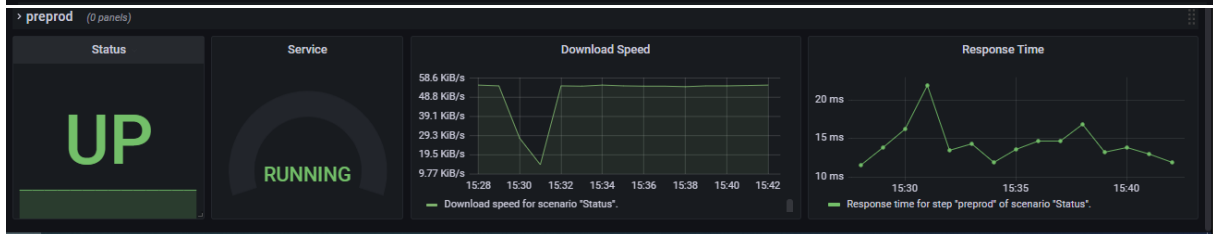
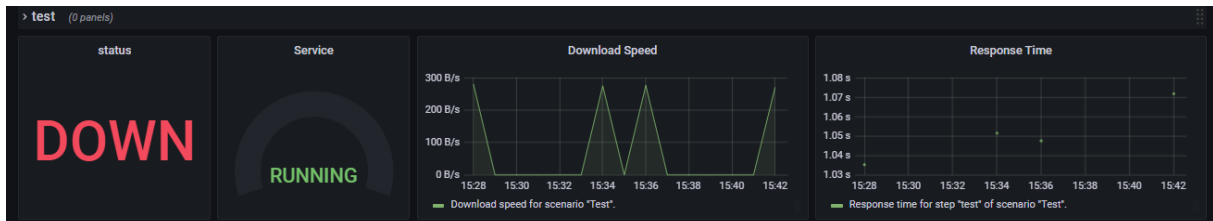
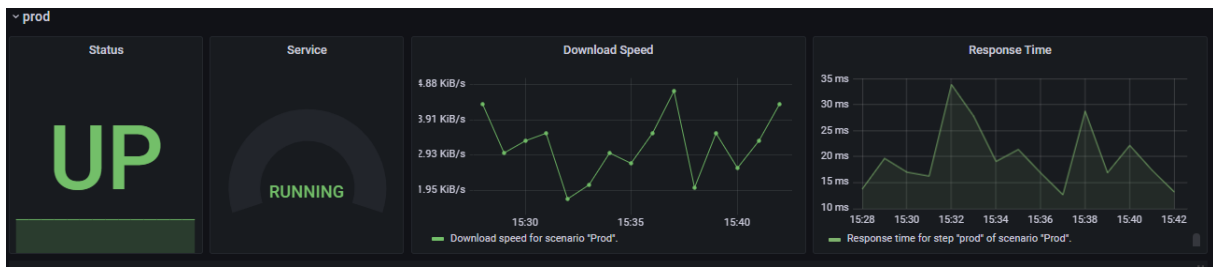
Show details ☐ Show items without data ☐

Host	Name	Last check	Last value	Change	Tags
<input type="checkbox"/> preprod	Download speed for scenario "Status".	2021-07-10 13:48:06	54.27 KBps		Graph
<input type="checkbox"/> preprod	Download speed for step "preprod" of scenario "Status".	2021-07-10 13:48:06	4.04 KBps		Graph
<input type="checkbox"/> preprod	Download speed for step "preproddeneme" of scenario "Status".	2021-07-10 13:48:06	104.49 KBps		Graph
<input type="checkbox"/> preprod	Failed step of scenario "Status".	2021-07-10 13:48:06	2		Graph
<input type="checkbox"/> preprod	HTTP service is running	2021-07-10 13:48:00	Up (1)	Application: HTTP ser...	Graph
<input type="checkbox"/> preprod	HTTPS service is running	2021-07-10 13:48:01	Down (0)	Application: HTTPS se...	Graph
<input type="checkbox"/> preprod	Last error message of scenario "Status".	2021-07-10 13:48:06	response code "503" ...		History
<input type="checkbox"/> preprod	Response code for step "preprod" of scenario "Status".	2021-07-10 13:48:06	200		Graph
<input type="checkbox"/> preprod	Response code for step "preproddeneme" of scenario "Status".	2021-07-10 13:48:06	503		Graph
<input type="checkbox"/> preprod	Response time for step "preprod" of scenario "Status".	2021-07-10 13:48:06	14.62ms		Graph
<input type="checkbox"/> preprod	Response time for step "preproddeneme" of scenario "Status".	2021-07-10 13:48:06	1.21ms		Graph

Displaying 11 of 11 found

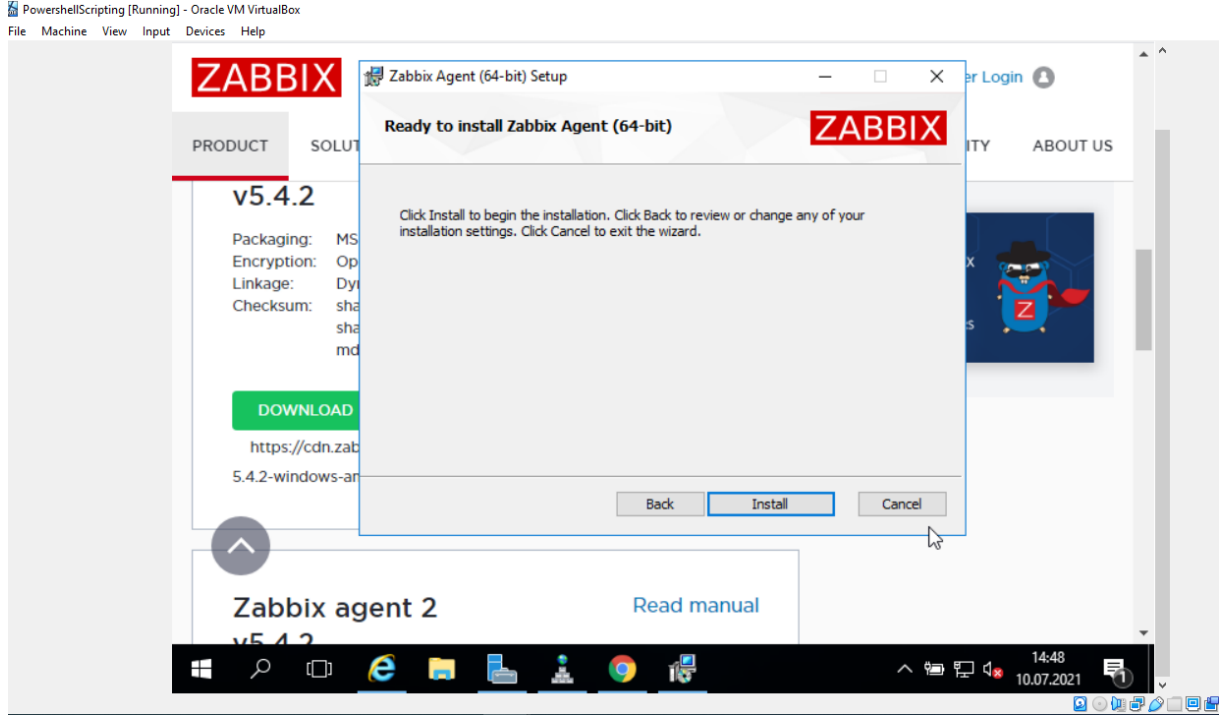


<input type="checkbox"/> prod	Download speed for scenario "Prod".	2021-07-10 14:22:15	3.54 KBps	+725 Bps	Graph
<input type="checkbox"/> preprod	Download speed for scenario "Status".	2021-07-10 14:22:09	27.7 KBps	-26.44 KBps	Graph
<input type="checkbox"/> test	Download speed for scenario "Test".	2021-07-10 14:21:53	283 Bps	+283 Bps	Graph
<input type="checkbox"/> preprod	Download speed for step "preprod" of scenario "Status".	2021-07-10 14:22:09	3.15 KBps	-644 Bps	Graph
<input type="checkbox"/> preprod	Download speed for step "preproddeneme" of scenario "Status".	2021-07-10 14:22:09	52.25 KBps	-52.25 KBps	Graph
<input type="checkbox"/> prod	Download speed for step "prod" of scenario "Prod".	2021-07-10 14:22:15	3.54 KBps	+725 Bps	Graph
<input type="checkbox"/> test	Download speed for step "test" of scenario "Test".	2021-07-10 14:21:53	283 Bps		Graph
<input type="checkbox"/> prod	Failed step of scenario "Prod".	2021-07-10 14:22:15	0		Graph
<input type="checkbox"/> preprod	Failed step of scenario "Status".	2021-07-10 14:22:09	2		Graph
<input type="checkbox"/> test	Failed step of scenario "Test".	2021-07-10 14:21:53	1		Graph
<input type="checkbox"/> prod	HTTP service is running	2021-07-10 14:21:18	Up (1)	Application: HTTP ser...	Graph
<input type="checkbox"/> preprod	HTTP service is running	2021-07-10 14:22:00	Up (1)	Application: HTTP ser...	Graph
<input type="checkbox"/> test	HTTP service is running	2021-07-10 14:22:11	Up (1)	Application: HTTP ser...	Graph



DNS Monitoring

DNS Server Sunucusuna (192.168.31.171) Zabbix Agent yüklendi. Agent ayarlarında aynı ağ üzerinde bulunan Agent Server IP adresi verildi (192.168.31.130).

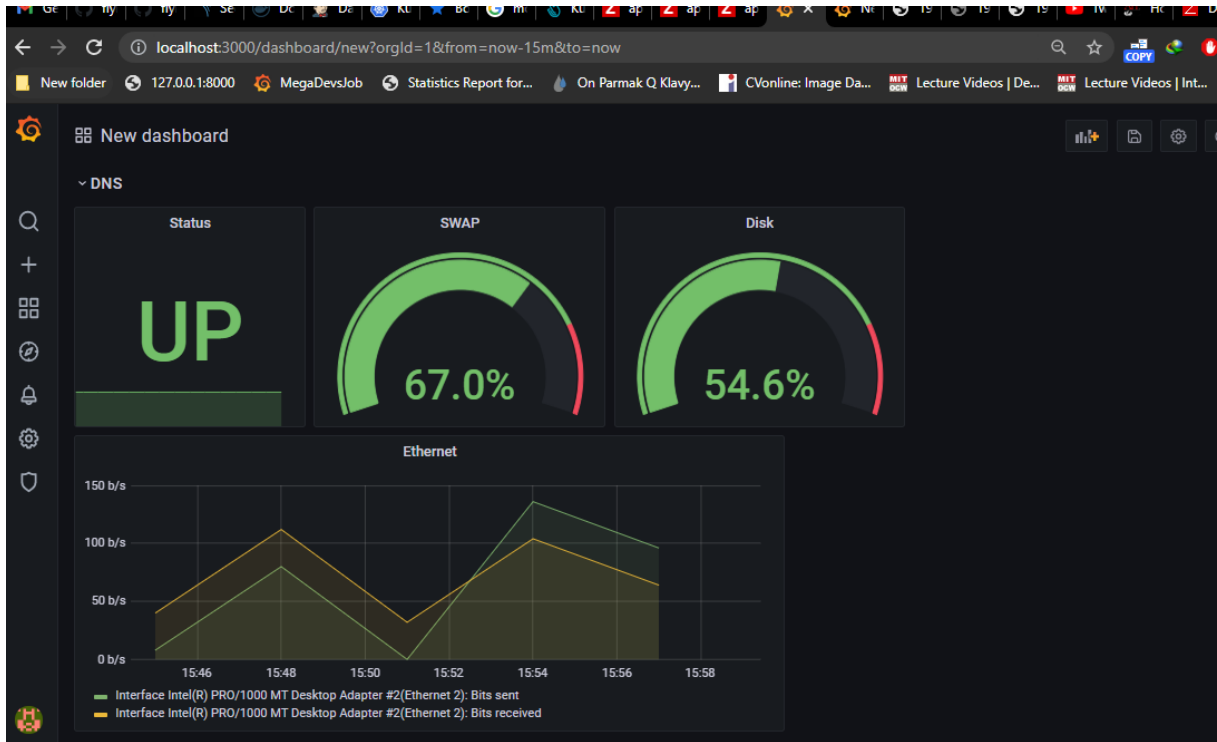


Zabbix üzerine entegrasyonu gerçekleştirildi.

192.168.31.98	192.168.31.97:10050	ZBX	13	Enabled	Latest data	Problems 13	Graphs 18	Dashboards 2	Web
DNS3	192.168.31.171:10050	ZBX		Enabled	Latest data	Problems	Graphs 7	Dashboards 2	Web
MegaDevsJob	127.0.0.1:10050	ZBX		Enabled	Latest data	Problems	Graphs	Dashboards	Web 2
preprod	192.168.31.101:30080	ZBX	1	Enabled	Latest data	Problems 1	Graphs	Dashboards	Web 1
prod	192.168.31.101:30082	ZBX		Enabled	Latest data	Problems	Graphs	Dashboards	Web 1
test	192.168.31.101:30081	ZBX		Enabled	Latest data	Problems	Graphs	Dashboards	Web 1
URL Monitoring	127.0.0.1:10050	ZBX	1	Enabled	Latest data	Problems 1	Graphs	Dashboards	Web 1
Zabbix server	127.0.0.1:10050	ZBX		Enabled	Latest data	Problems	Graphs 27	Dashboards 3	Web

Displaying 8 of 8 found

Grafana üzerinden DNS Sunucusu için bir panel oluşturuldu.



1.10 Ansible ile rsyslog, netsus ve tcpdump kurulumu

Ansible server 192.168.31.101 numaralı sunucu üzerinde kurulup 192.168.31.205 numaralı Jenkins sunucusu ile ssh bağlantısı gerçekleştirildi.

Rsyslog kurulumu

Host dosyasının içeriğinde sunucuların IP bilgileri belirtilmiştir. Rsyslog kurulumu için gerekli aşamaları içeren ansible playbook dosyası aşağıdaki gibidir.

```
[all]
kubespary ansible host="192.168.31.101"
busra ansible_host="192.168.31.205"
```

```
- name: my playbook1
  hosts: all
  become: true
  become_method: sudo
  become_user: root
  remote_user: busra
  tasks:
    - name: rsyslog serverı kurma Redhat
      yum:
        name: rsyslog
        state: latest
        when: ansible_os_family=='RedHat'
```

Playbook çalıştırılır ve rsyslog kurulumu yapıldığına dair kontrol edilir.

```
root@master:~/ansible/playbook_example# ansible-playbook -i host.ini playbooks/playbook1.yaml -K
```

```
PLAY [my playbook1] *****
TASK [Gathering Facts] *****
ok: [kubespray]
ok: [busra]

TASK [rsyslog serverı kurma Redhat] *****
skipping: [kubespray]
skipping: [busra]

TASK [rsyslog serverı kurma Debian] *****
ok: [kubespray]
ok: [busra]

TASK [rsyslog service restart] *****
changed: [kubespray]
changed: [busra]
```

```
busra@busra:~$ systemctl status rsyslog.service
● rsyslog.service - System Logging Service
   Loaded: loaded (/lib/systemd/system/rsyslog.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2021-07-10 16:02:16 UTC; 29min ago
     Docs: man:rsyslogd(8)
           http://www.rsyslog.com/doc/
   Main PID: 4092 (rsyslogd)
     Tasks: 4 (limit: 4655)
    CGroup: /system.slice/rsyslog.service
            └─4092 /usr/sbin/rsyslogd -n
```

Netstat Kurulumu

Netstat komutunu içeren net-tools paketini kurmak amacıyla aşağıdaki playbook çalıştırılmıştır. Sonrasında kurulumun status durumunu kontrol etmek amacıyla versiyon bilgileri Output olarak ekran üzerinde belirtilmiştir.

```

---
- name: netstat
  hosts: all
  become: true
  become_method: sudo
  become_user: root
  remote_user: busra
  tasks:
    - name: netstat installation on RedHat
      yum:
        name: net-tools
        state: latest
        when: ansible_os_family=='RedHat'

    tasks:
      - name: netstat installation on RedHat
        apt:
          name: net-tools
          state: latest
          when: ansible_os_family=='Debian'

    tasks:
      - name: check netstat version
        shell: "netstat --version"
        args:
          executable: "/bin/bash"
        register: nv

      - name: output
        debug:
          msg: "{{ nv.stdout_lines }}"

```

Output çıktısı aşağıdaki gibidir.

```

TASK [check netstat version] *****
changed: [kubespri]
changed: [busra]

TASK [output] *****
ok: [kubespri] => {
  "msg": [
    "net-tools 2.10-alpha",
    "Fred Baumgarten, Alan Cox, Bernd Eckenfels, Phil Blundell, Tuan Hoang, Brian Micek and others",
    "+NEW ADDRT +RTF IRTT +RTF REJECT +FW MASQUERADE +I18N +SELINUX",
    "AF: (inet) +UNIX +INET +INET6 +IPX +AX25 +NETROM +X25 +ATALK +ECONET +ROSE -BLUETOOTH",
    "HW: +ETHER +ARC +SLIP +PPP +TUNNEL -TR +AX25 +NETROM +X25 +FR +ROSE +ASH +SIT +FDDI +HIPPI +HDLCP/LAPB +EUI64"
  ]
}
ok: [busra] => {
  "msg": [
    "net-tools 2.10-alpha",
    "Fred Baumgarten, Alan Cox, Bernd Eckenfels, Phil Blundell, Tuan Hoang, Brian Micek and others",
    "+NEW ADDRT +RTF IRTT +RTF REJECT +FW MASQUERADE +I18N +SELINUX",
    "AF: (inet) +UNIX +INET +INET6 +IPX +AX25 +NETROM +X25 +ATALK +ECONET +ROSE -BLUETOOTH",
    "HW: +ETHER +ARC +SLIP +PPP +TUNNEL -TR +AX25 +NETROM +X25 +FR +ROSE +ASH +SIT +FDDI +HIPPI +HDLCP/LAPB +EUI64"
  ]
}

```

Tcpdump playbook aracılığıyla kurulur ve test edilir.

```

---
- name: tcpdump
  hosts: all
  become: true
  become_method: sudo
  become_user: root
  remote_user: busra
  tasks:
    - name: tcpdump installation on RedHat
      yum:
        name: tcpdump
        state: latest
        when: ansible_os_family=='RedHat'

    tasks:
      - name: tcpdump installation on Debian
        apt:
          name: tcpdump
          state: latest
          when: ansible_os_family=='Debian'

    tasks:
      - name: tcpdump shell
        shell: "tcpdump -D"
        args:
          executable: "/bin/bash"
        register: tdump

      - name: output
        debug:
          msg: "{{ tdump.stdout_lines }}"

```

```

TASK [tcpdump shell] *****
changed: [busra]
changed: [kubespray]

TASK [output] *****
ok: [kubespray] => {
  "msg": [
    "1.docker0 [Up, Running]",
    "2.datapath [Up, Running]",
    "3.weave [Up, Running]",
    "4.vethwe-datapath [Up, Running]",
    "5.vethwe-bridge [Up, Running]",
    "6.vethwepl1bc6cf0 [Up, Running]",
    "7.vethwepldbac7a1 [Up, Running]",
    "8.enp0s3 [Up, Running]",
    "9.vethwepl33fba5e [Up, Running]",
    "10.enp0s8 [Up, Running]",
    "11.vethwepl9b58f24 [Up, Running]",
    "12.br-1c186d178e26 [Up, Running]",
    "13.vethwepl23ddb28 [Up, Running]",
    "14.vethbcbfed05 [Up, Running]",
    "15.vethweplbb74e93 [Up, Running]",
    "16.vethweplc5a6e95 [Up, Running]",
    "17.vethbfcf99f [Up, Running]",

```

Kubernetes ve Docker version bilgileri kubescape sunucusu üzerinden elde edilir.

```
hosts: kubescape
become: true
become_method: sudo
become_user: root
remote_user: busra
tasks:
- name: kubernetes version
  shell: "kubectl version"
  args:
    executable: "/bin/bash"
  register: kv

- name: output
  debug:
    msg: "{{ kv.stdout_lines }}"

- name: versiondocker
  hosts: kubescape
  become: true
  become_method: sudo
  become_user: root
  remote_user: busra
  tasks:
  - name: docker version
    shell: "docker version"
    args:
      executable: "/bin/bash"
    register: dv

  - name: output
    debug:
      msg: "{{ dv.stdout_lines }}"
```



```
TASK [output] *****
ok: [kubespray] => {
  "msg": [
    "Client: Docker Engine - Community",
    " Version:          20.10.7",
    " API version:       1.41",
    " Go version:        go1.13.15",
    " Git commit:        f0df350",
    " Built:             Wed Jun  2 11:56:40 2021",
    " OS/Arch:           linux/amd64",
    " Context:           default",
    " Experimental:      true",
    "",
    "Server: Docker Engine - Community",
    " Engine:",
    " Version:          20.10.7",
    " API version:       1.41 (minimum version 1.12)",
    " Go version:        go1.13.15",
    " Git commit:        b0f5bc3",
    " Built:             Wed Jun  2 11:54:48 2021",
    " OS/Arch:           linux/amd64",
    " Experimental:      false",
    " containerd:",
    " Version:          1.4.6",
    " GitCommit:        d71fcd7d8303cbf684402823e425e9dd2e99285d",
    " runc:",
    " Version:          1.0.0-rc95",
    " GitCommit:        b9ee9c6314599f1b4a7f497elf1f856fe433d3b7",
    " docker-init:",
    " Version:          0.19.0",
    " GitCommit:        de40ad0"
  ]
}
```

```
TASK [output] *****
ok: [kubespray] => {
  "msg": [
    "Client Version: version.Info{Major:\"1\", Minor:\"20\", GitVersion:\"v1.20.4\", GitCommit:\"e87da0bd6e03ec3fea7933c4b5263d151aafd07c\", GitTreeState:\"clean\", BuildDate:\"2021-02-18T16:12:00Z\", GoVersion:\"go1.15.8\", Compiler:\"gc\", Platform:\"linux/amd64\"}",
    "Server Version: version.Info{Major:\"1\", Minor:\"20\", GitVersion:\"v1.20.4\", GitCommit:\"e87da0bd6e03ec3fea7933c4b5263d151aafd07c\", GitTreeState:\"clean\", BuildDate:\"2021-02-18T16:03:00Z\", GoVersion:\"go1.15.8\", Compiler:\"gc\", Platform:\"linux/amd64\"}"
  ]
}
```

1.11 OWASP

Owasp üzerinden altyapı açıklarını analiz etmek amacıyla ilk olarak Kali kuruldu ve sonrasında ilgili NodePort üzerinden bir attack işlemi gerçekleştirildi.

The screenshot shows the OWASP Quick Start web interface. It has a navigation bar with 'Quick Start', 'Request', and 'Response' tabs. The main content area contains instructions to launch an automated scan by entering a URL. Below the instructions, there is a form with the following fields and options:

- URL to attack:** A text input field containing 'http://192.168.31.101:30080' and a 'Select...' button.
- Use traditional spider:** An unchecked checkbox.
- Use ajax spider:** A checked checkbox with a dropdown menu set to 'Firefox'.
- Buttons:** 'Attack' (with a lightning bolt icon) and 'Stop' (with a square icon).
- Progress:** A status message that reads 'Attack complete - see the Alerts tab for details of any issues found'.

The screenshot shows the Burp Suite interface. The top bar includes tabs for History, Search, Alerts, Output, Spider, Active Scan, and AJAX Spider. The Active Scan tab is selected, showing a progress bar at 40% and a list of sent messages. Below this, a table displays the results of the scan, including request timestamps, response timestamps, methods, URLs, codes, reasons, RTT, and response sizes. The table shows four requests, all with a 404 NOT FOUND status. The Alerts tab is also visible, showing two alerts: 'X-Frame-Options Header Not Set (2)' and 'X-Content-Type-Options Header Missing (2)'. The solution for the first alert is provided: 'site (if you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. Alternatively consider'.

Id	Req. Timestamp	Resp. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Header	Size Resp. Body
106	7/13/21, 3:02:43 PM	7/13/21, 3:02:43 PM	GET	http://192.168.31.101:30080/	200	OK	24 ...	154 bytes	58 bytes
108	7/13/21, 3:02:43 PM	7/13/21, 3:02:43 PM	GET	http://192.168.31.101:30080/favicon.ico	404	NOT FOUND	14 ...	162 bytes	232 bytes
107	7/13/21, 3:02:43 PM	7/13/21, 3:02:43 PM	GET	http://192.168.31.101:30080/robots.txt	404	NOT FOUND	8 ms	162 bytes	232 bytes
109	7/13/21, 3:02:44 PM	7/13/21, 3:02:44 PM	GET	http://192.168.31.101:30080/sitemap.xml	404	NOT FOUND	20 ...	162 bytes	232 bytes

1.12 Powershell Scripting

Windows üzerindeki bir servisi incelemek amacıyla ilk olarak slave konumunda Windows node eklendi.

The screenshot shows the Jenkins interface. The top section displays a table of nodes, including 'master' (Linux amd64) and 'win' (Windows 10 amd64). The 'win' node is highlighted with a red box. Below the table, the configuration for the 'win' node is shown, including fields for Name, Description, Number of executors, Remote root directory, Labels, and Usage.

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	master	Linux (amd64)	In sync	10.11 GB	2.00 GB	10.11 GB	0ms
	win	Windows 10 (amd64)	2 sec behind	17.80 GB	15.10 GB	17.80 GB	211ms
	winremote		N/A	N/A	N/A	N/A	N/A

Data obtained: 44 min, 44 min, 44 min, 44 min, 44 min, 44 min, 44 min

Refresh status

Name: win

Description: PowershellScript2

Number of executors: 2

Remote root directory: C:\Users\MYK\Desktop

Labels: PowershellScript2

Usage: Only build jobs with label expressions matching this node

Launch agent by connecting it to the master

☐ Disable WorkDir

Custom WorkDir path

Internal data directory

remoting

☐ Fail if workspace is missing

☒ Use WebSocket

Advanced...

Availability

Keep this agent online as much as possible

Node Properties

☐ Disable deferred wipeout on this node

☐ Environment variables

☐ Tool Locations

Save

İzlenecek sunucu üzerinde jenkins agent kuruldu ve bağlantı aktif hale getirildi.

```
PS C:\Users\Administrator\Downloads> java -jar agent.jar -jnlpUrl http://192.168.31.205:8080/computer/winr
java : Tem 12, 2021 2:03:33 PM org.jenkinsci.remoting.engine.WorkDirManager initializeWorkDir
At line:1 char:1
+ java -jar agent.jar -jnlpUrl http://192.168.31.205:8080/computer/winr ...
+ ~~~~~
+ CategoryInfo          : NotSpecified: (Tem 12, 2021 2:...initializeWorkDir:String) [], RemoteExce
+ FullyQualifiedErrorId : NativeCommandError

INFO: Using C:\Users\Administrator\Desktop\remoting as a remoting work directory
Tem 12, 2021 2:03:33 PM org.jenkinsci.remoting.engine.WorkDirManager setupLogging
INFO: Both error and output logs will be printed to C:\Users\Administrator\Desktop\remoting
Tem 12, 2021 2:03:34 PM hudson.remoting.jnlp.Main createEngine
INFO: Setting up agent: winremote
Tem 12, 2021 2:03:34 PM hudson.remoting.jnlp.Main$CuiListener <init>
INFO: Jenkins agent is running in headless mode.
Tem 12, 2021 2:03:34 PM hudson.remoting.Engine startEngine
INFO: Using Remoting version: 4.7
Tem 12, 2021 2:03:34 PM org.jenkinsci.remoting.engine.WorkDirManager initializeWorkDir
INFO: Using C:\Users\Administrator\Desktop\remoting as a remoting work directory
Tem 12, 2021 2:03:35 PM hudson.remoting.jnlp.Main$CuiListener status
INFO: WebSocket connection open
Tem 12, 2021 2:03:40 PM hudson.remoting.jnlp.Main$CuiListener status
INFO: Connected
```



Agent win (PowershellScript2)

Agent is connected.

Labels

PowershellScript2

Projects tied to win

None

Yeni bir pipeline oluşturup label üzerinden hangi sunucu üzerinde çalıştırılacağı (win) belirtildi.

☒ Restrict where this project can be run

Label Expression

PowershellScript

Label PowershellScript matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.

Advanced...

Sunucuyu her saniye kontrol etmesi sağlandı.

Build Triggers

☐ Trigger builds remotely (e.g., from scripts)

☐ Build after other projects are built

☐ Build periodically

☐ GitHub hook trigger for GITScm polling

☒ Poll SCM

Schedule

Do you really mean "every minute" when you say "*****"? Perhaps you meant "H*****" to poll once per

Credential Bilgileri girildi.

Username Variable ?

USERNAME

Password Variable ?

PASSWORD

Credentials ?

☒ Specific credentials ☐ Parameter expression

Administrator/***** (POWERSHELL3) Add

Powershell Script yazıldı ve bu kısımda notepad process durumuna bakıldı, process içerisinde olmaması durumunda başlatıldı.

```
PowerShell
Command
1 $PASSWORD = ConvertTo-SecureString "$($ENV:PASSWORD)" -AsPlainText -Force
2 $Credential = New-Object System.Management.Automation.PSCredential ("$ENV:USERNAME", $PASSWORD)
3 $service = Get-Service ALG -ErrorAction SilentlyContinue
4 if ($service.status -eq "stopped")
5 {
6     Start-Service -Name ALG | where ($_.credential -eq $Credential)
7 }
8
```

Output aşağıdaki gibidir.

 **Console Output**

Started by timer
Running as SYSTEM
Building remotely on win (PowershellScript2) in workspace C:\Users\MYK\Desktop\workspace\PowershellScript
[PowershellScript] \$ powershell.exe -NonInteractive -ExecutionPolicy Bypass -File C:\Users\MYK\AppData\Local\Temp\jenkins8090014713899497724.ps1
Finished: SUCCESS

```
PS C:\WINDOWS\system32> Get-Service -Name alg

Status  Name                DisplayName
-----
Running alg            Uygulama Katmanı Ağ Geçidi Hizmeti

PS C:\WINDOWS\system32> Stop-Service -Name alg
PS C:\WINDOWS\system32> Get-Service -Name alg

Status  Name                DisplayName
-----
Stopped alg            Uygulama Katmanı Ağ Geçidi Hizmeti

PS C:\WINDOWS\system32> Get-Service -Name alg

Status  Name                DisplayName
-----
Running alg            Uygulama Katmanı Ağ Geçidi Hizmeti
```

Sunucu üzerinde bulunan Disk Boyutları HTML formatında raporlandı ve Credential ve STMP kullanarak mail olarak atıldı.

```
PS C:\WINDOWS\system32> Get-Volume | Where-Object {$PSItem.DriveType -eq "Fixed"} | ConvertTo-Html | Out-File -FilePath C:\volume.html
$From = "killchusrai3@gmail.com"
$To = "killchusrai3@gmail.com"
$Attachment = "C:\volume.html"
$Subject = "Disk Volumes"
$Body = "Disk boyutlarını içeren html dosyası ektedir"
$SMTPServer = "smtp.gmail.com"
$SMTPPort = "587"
Send-MailMessage -From $From -to $To -Subject $Subject -Body $Body -SmtpServer $SMTPServer -port $SMTPPort -UseSsl -Credential (Get-Credential) -Attachments $Attachment
cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
```

Output ve rapor içerikleri aşağıdaki gibidir.

OperationalStatus	HealthStatus	DriveType	FileSystemType	DedupMode	ObjectId	PassThroughClass	PassThroughId	PassThroughNamespace	PassThroughServer	UniqueId	AllocationUnitSize	Drive
OK	Healthy	Fixed	NTFS	NotAvailable	(1) (DESKTOP-B9NQK6 root:Microsoft Windows Storage Providers_v2\WSP_Volume ObjectId="(d6fa1b9-a180-11e8-b91a-80e6f6e9963)\Volume{(0848a398-8dad-4a79-b278-c0a6fec3ba9d)})"					Volume{(0848a398-8dad-4a79-b278-c0a6fec3ba9d)}	4096	C
OK	Healthy	Fixed	NTFS	NotAvailable	(1) (DESKTOP-B9NQK6 root:Microsoft Windows Storage Providers_v2\WSP_Volume ObjectId="(d6fa1b9-a180-11e8-b91a-80e6f6e9963)\Volume{(8871ac23-b251-4c9e-847d-81b90a15d912)})"					Volume{(8871ac23-b251-4c9e-847d-81b90a15d912)}	4096	
OK	Healthy	Fixed	NTFS	NotAvailable	(1) (DESKTOP-B9NQK6 root:Microsoft Windows Storage Providers_v2\WSP_Volume ObjectId="(d6fa1b9-a180-11e8-b91a-80e6f6e9963)\Volume{(95ea1490-0567-4676-83c7-7043ce3e82dc)})"					Volume{(95ea1490-0567-4676-83c7-7043ce3e82dc)}	4096	D
OK	Healthy	Fixed	FAT32	NotAvailable	(1) (DESKTOP-B9NQK6 root:Microsoft Windows Storage Providers_v2\WSP_Volume ObjectId="(d6fa1b9-a180-11e8-b91a-80e6f6e9963)\Volume{(d8764e9d-c57a-45ab-96e2-277b6c614970)})"					Volume{(d8764e9d-c57a-45ab-96e2-277b6c614970)}	1024	
OK	Healthy	Fixed	NTFS	NotAvailable	(1) (DESKTOP-B9NQK6 root:Microsoft Windows Storage Providers_v2\WSP_Volume ObjectId="(d6fa1b9-a180-11e8-b91a-80e6f6e9963)\Volume{(a9bc2c85-de32-4751-988c-0b92ce5d5b7)})"					Volume{(a9bc2c85-de32-4751-988c-0b92ce5d5b7)}	4096	
OK	Healthy	Fixed	FAT32	NotAvailable	(1) (DESKTOP-B9NQK6 root:Microsoft Windows Storage Providers_v2\WSP_Volume ObjectId="(d6fa1b9-a180-11e8-b91a-80e6f6e9963)\Volume{(abc3def-fd8-4c2a-b3b8-c0a5e931527)})"					Volume{(abc3def-fd8-4c2a-b3b8-c0a5e931527)}	4096	

Web API üzerinden GET isteği için <http://open-notify.org/> üzerinde ISS location bilgilerini bulunduran API kullanıldı.

```
#api üzerinde get ve post işlemleri
$sp=Invoke-RestMethod -Method GET -Uri http://api.open-notify.org/iss-now.json
$sp.iss_position
```

```
PS C:\WINDOWS\system32> $sp=Invoke-RestMethod -Method GET -Uri http://api.open-notify.org/iss-now.json
$sp.iss_position

latitude longitude
-----
-25.2051 84.5167
```

POST API için <http://jsonplaceholder.typicode.com/todos> adresin de bulunan API kullanıldı.

```
#Post İşlemi
$ToDo = @{
    title= "Finish TBB Final Project";
    completed="True";
}
$json = $ToDo | ConvertTo-Json
Invoke-RestMethod -Method Post -Uri "http://jsonplaceholder.typicode.com/todos" -Body $json -ContentType 'application/json'
```

```
PS C:\WINDOWS\system32> $ToDo = @{
    title= "Finish TBB Final Project";
    completed="True";
}
$json = $ToDo | ConvertTo-Json
Invoke-RestMethod -Method Post -Uri "http://jsonplaceholder.typicode.com/todos" -Body $json -ContentType 'application/json'

completed title id
-----
True Finish TBB Final Project 201
```

