Программное обеспечение «Track & Trace System»

ИНСТРУКЦИЯ ПО УСТАНОВКЕ

версия 1.0.1

Листов 12

2024

АННОТАЦИЯ

В настоящем документе представлена Инструкция по установке программного обеспечения «Track & Trace System» (далее – ПО). Контакты технических специалистов, которые могут проконсультировать по процессу развёртывания и настройки экземпляра ПО и его функционирования: Аблязов Марат, m.ablyazov@crpt.ru, тел.: 89161286727.

1. УСТАНОВКА ПО

1.1. Общие сведения по установке ПО

Разработка ПО велась в соответствии с практиками «непрерывная интеграция/непрерывное развертывание» (CI/CD, Continuous Integration/Continuous Delivery).

Разработка ПО велась с использованием инструмента GitLab, который имеет встроенную систему контроля версий и позволяет выполнять совместную разработку силами нескольких команд, применять обновления кода, выполнять сборку и при необходимости откатывать изменения.

Сборка релизов происходит в среде непрерывной интеграции и доставки GitLab CI/CD в соответствии со скриптом (сценарием) сборки приложения. Также развернута тестовая среда (так называемый, контур разработки и тестирования), в которой происходит развертывание разработанных компонентов программного обеспечения, их автономное и совместное тестирование.

Для доставки ПО в промышленную или тестовую среду заказчика необходимо иметь docker-образы сервисов ПО и развернуть их в контур заказчика. Предполагается развертывание ПО в составе общей сборки с компонентами прикладной инфраструктуры и технологическими компонентами.

Развертывание ПО выполняется в следующей последовательности:

1) общая подготовка серверов;
2) установка кластера Kubernetes;
3) установка docker;
4) установка docker-registry;
5) установка rancher;
6) инициализация кластера Kubernetes;
7) получение kubeconfig;
8) развертывание системных сервисов в кластер;
9) развертывание компонентов системы мониторинга;
10) установка компонентов системы логирования;
11) развертывание и настройка кластера СУБД PostgreSQL.

1.2. Общая подготовка серверов

Установка ПО производится на предустановленную операционную систему Ubuntu Server 22.04.

Для настройки ПО используется утилита ansible версии не ниже 2.13.

1.3. Установка кластера Kubernetes

Создать inventory-файл с указанием ip-адресов виртуальных машин, следующего содержания:

```
all:
  vars:
    domain: "itobacco.i-tob.net"
    dnsServers:
      - 10.179.3.71
      - 10.179.3.72
    ntp-server:
      - 10.179.3.71
      - 10.179.3.72
  hosts:
    docker-registry:
      ansible_host: 10.179.51.5
    k8s-master-1:
      ansible_host: 10.179.51.10
    k8s-master-2:
      ansible_host: 10.179.51.11
    k8s-master-3:
      ansible_host: 10.179.51.12
    elastic-1:
      ansible_host: 10.179.51.21
    elastic-2:
      ansible_host: 10.179.51.22
    elastic-3:
      ansible_host: 10.179.51.23
    k8s-worker-1:
      ansible_host: 10.179.51.31
    k8s-worker-2:
      ansible_host: 10.179.51.32
    k8s-worker-3:
      ansible_host: 10.179.51.33
    victoria-1:
```

```
    ansible_host: 10.179.51.41
  victoria-2:
    ansible_host: 10.179.51.42
  victoria-3:
    ansible_host: 10.179.51.43
  data-exchange-1:
    ansible_host: 10.179.51.61
  data-exchange-2:
    ansible_host: 10.179.51.62
  data-exchange-3:
    ansible_host: 10.179.51.63
children:
  k8s-cluster:
    vars:
      k8s_labels: []
      k8s_taints: []
    children:
      kube-node:
        hosts:
          k8s-worker-1:
          k8s-worker-2:
          k8s-worker-3:
      kube-master:
        hosts:
          k8s-master-1:
          k8s-master-2:
          k8s-master-3:
      logging-nodes:
        hosts:
          elastic-1:
          elastic-2:
          elastic-3:
      mq-nodes:
        hosts:
          data-exchange-1:
```

```
        data-exchange-2:
        data-exchange-3:
      monitoring-nodes:
        hosts:
          victoria-1:
          victoria-2:
          victoria-3:
docker-nodes:
  hosts:
    docker-registry:
    k8s-master-1:
    k8s-master-2:
    k8s-master-3:
    elastic-1:
    elastic-2:
    elastic-3:
    k8s-worker-1:
    k8s-worker-2:
    k8s-worker-3:
    victoria-1:
    victoria-2:
    victoria-3:
    data-exchange-1:
    data-exchange-2:
    data-exchange-3:
docker-registries:
  hosts:
    docker-registry:
rancher:
  hosts:
docker-registry:
```

1.4. Установка docker

В рабочей папке ansible запустить ansible-playbook с следующими параметрами:

```
ansible-playbook -i environment/prod/inventory.yaml \
```

```
playbooks/service/docker.yaml \
playbooks/service/timesyncd.yaml \
-u %USER_NAME%
```

Значение %USER_NAME% - имя пользователя, с привилегиями администратора, от которого производятся операции в системе виртуальной машины.

1.5. Установка docker-registry

В рабочей папке ansible запустить ansible-playbook с следующими параметрами:

```
ansible-playbook -i environment/prod/inventory.yaml \
playbooks/service/docker-registry.yaml -u %USER_NAME%
```

Значение %USER_NAME% - имя пользователя, с привилегиями администратора, от которого производятся операции в системе виртуальной машины.

1.6. Установка Rancher

В файле environment/prod/host_vars/rancher/rancher.yaml изменить значение переменной rancher_datadir на путь до папки хранения данных Rancher.

В рабочей папке ansible запустить ansible-playbook с следующими параметрами:

```
ansible-playbook -i environment/prod/inventory.yaml \
playbooks/service/rancher.yaml -u %USER_NAME%
```

Значение %USER_NAME% - имя пользователя, с привилегиями администратора, от которого производятся операции в системе виртуальной машины.

1.7. Инициализация кластера Kubernetes

1) После установки Rancher открыть веб интерфейс https://%RANCHER_HOST%:8443/g/clusters;
2) Нажать «Add cluster»;
3) Выбрать «Existing nodes»;
4) Установить следующие значения параметров:

```
Cluster Name: itobacco
    Kubernetes Version: v1.21
    Network Provider: Calico
    Advanced options:
        Nginx Ingress: Disabled
        Node Port Range: 25000-35000
        Metrics Server Monitoring: Disabled
```

5) Далее нажать «Next»;

6) Скопировать значения server, token, ca-checksum и вписать их в файл environment/prod/group_vars/k8s-cluster/rancher.yaml:

```
---
rancher_host: "%server%"
rancher_token: "%token%"
rancher_ca_checksum: "%ca-checksum%"
```

7) В рабочей папке ansible запустить ansible-playbook с следующими параметрами:

```
ansible-playbook -i environment/prod/inventory.yaml \
        playbooks/rancher-node.yaml -u %USER_NAME%
```

Значение %USER_NAME% - имя пользователя, с привилегиями администратора, от которого производятся операции в системе виртуальной машины;

8) Дождаться инициализации кластера Kubernetes.

1.8. Получение kubeconfig

1) Открыть веб интерфейс https://%RANCHER_HOST%:8443/g/clusters;
2) Выбрать кластер;
3) Нажать «Kubeconfig File» и сохранить содержимое в домашнюю директорию в файл .kube/config.

1.9. Установка системных сервисов в кластер

Выполнить следующие команды:

```
# CertManager
helm -n cert-manager --create-namespace install cert-manager \
 ./k8s/helm/charts/cert-manager -f\
 ./k8s/helm/values/prod/cert-manager/values.yaml
# Ingress controller
helm -n ingress-nginx --create-namespace install nginx \
 ./k8s/helm/charts/ingress-nginx \
 -f ./k8s/helm/values/prod/ingress-nginx/values.yaml
# Load Balancer
helm -n metallb-system --create-namespace install metallb \
 ./k8s/helm/charts/metallb \
 ./k8s/helm/values/prod/metallb/values.yaml
helm -n metallb-system install extra-manifests \
 ./k8s/helm/charts/metallb \
 ./k8s/helm/values/prod/extra-manifests/metallb-system.yaml
```

```
# Storage controller
helm -n kube-system install csi-driver-smb \
 ./k8s/helm/charts/csi-driver-smb \
 ./k8s/helm/values/prod/csi-driver-smb/values.yaml
helm -n kube-system install local-path-provisioner \
 ./k8s/helm/charts/local-path-provisioner \
 ./k8s/helm/values/prod/local-path-provisioner/values.yaml
# RabbitMQ Operator
helm -n kube-system install rabbitmq-operator \
 ./k8s/helm/charts/rabbitmq-operator \
 ./k8s/helm/values/prod/rabbitmq-operator/values.yaml
```

1.10. Установка компонентов системы мониторинга

Выполнить следующие команды:

```
# VictoriaMetricsOperator &  VictoriaMetrics
helm -n monitoring --create-namespace install victoria-metrics \
 ./k8s/helm/charts/victoria-metrics \
 -f ./k8s/helm/values/prod/victoria-metrics/values.yaml
# GrafanaOprtator & Grafana
helm -n monitoring install grafana-operator \
 ./k8s/helm/charts/grafana-operator \
 -f ./k8s/helm/values/prod/grafana-operator/values.yaml
# Extra manifests
helm -n monitoring install extra-manifests \
 ./k8s/helm/charts/extra-manifests \
 -f ./k8s/helm/values/prod/extra-manifests/values-monitoring.yaml
# Dashboards & Rules & AlertRules
helm -n monitoring install monitoring-app \
 ./k8s/helm/charts/monitoring-app \
 -f ./k8s/helm/values/prod/monitoring/common.yaml \
 -f ./k8s/helm/values/prod/monitoring/monitoring-app.yaml
helm -n monitoring install monitoring-db \
 ./k8s/helm/charts/monitoring-db \
 -f ./k8s/helm/values/prod/monitoring/common.yaml \
 -f ./k8s/helm/values/prod/monitoring/monitoring-db.yaml
helm -n monitoring install monitoring-infra \
```

```
 ./k8s/helm/charts/monitoring-infra \
 -f ./k8s/helm/values/prod/monitoring/common.yaml \
 -f ./k8s/helm/values/prod/monitoring/monitoring-infra.yaml
helm -n monitoring install monitoring-k8s \
 ./k8s/helm/charts/monitoring-k8s \
 -f ./k8s/helm/values/prod/monitoring/common.yaml \
 -f ./k8s/helm/values/prod/monitoring/monitoring-k8s.yaml
helm -n monitoring install monitoring-mq \
 ./k8s/helm/charts/monitoring-mq \
 -f ./k8s/helm/values/prod/monitoring/common.yaml \
 -f ./k8s/helm/values/prod/monitoring/monitoring-mq.yaml
```

1.11. Установка компонентов системы логирования

Выполнить следующие команды:

```
# FluentBit
helm -n logging-system --create-namespace install fluent-bit \
 ./k8s/helm/charts/fluent-bit \
 -f ./k8s/helm/values/prod/fluent-bit/values.yaml
# OpenSearch
helm -n logging-system install opensearch-master \
 ./k8s/helm/charts/opensearch \
 -f ./k8s/helm/values/prod/opensearch/common.yaml \
 -f ./k8s/helm/values/prod/opensearch/opensearch-master.yaml
helm -n logging-system install opensearch-data \
 ./k8s/helm/charts/opensearch \
 -f ./k8s/helm/values/prod/opensearch/common.yaml \
 -f ./k8s/helm/values/prod/opensearch/opensearch-data.yaml
helm -n logging-system install opensearch-ingest \
 ./k8s/helm/charts/opensearch \
 -f ./k8s/helm/values/prod/opensearch/common.yaml \
 -f ./k8s/helm/values/prod/opensearch/opensearch-ingest.yaml
```

1.12. Развертывание кластера СУБД PostgreSQL

Создать inventory-файл с указанием ip-адресов виртуальных машин, следующего содержания:

```
all:
```

```
  vars:
    ansible_python_interpreter: '/usr/bin/python3'
    subnet_cidr: "10.179.50.0/23"
    nameservers:
      - 10.179.3.71
      - 10.179.3.72
    ntpservers:
      - 10.179.3.71
      - 10.179.3.72
  hosts:
    etcd-node-1:
      ansible_host: 10.179.51.71
    etcd-node-2:
      ansible_host: 10.179.51.72
    etcd-node-3:
      ansible_host: 10.179.51.73
    db-node-1:
      ansible_host: 10.179.51.81
    db-node-2:
      ansible_host: 10.179.51.82
    db-node-3:
      ansible_host: 10.179.51.83

etcdcluster:
  hosts:
    etcd-node-1:
    etcd-node-2:
    etcd-node-3:
balancers:
  hosts:
    db-node-1:
    db-node-2:
    db-node-3:
master:
  hosts:
```

```
    db-node-1:
replica:
  hosts:
    db-node-2:
    db-node-3:
pgcluster:
  children:
    master:
    replica:
```

В рабочей папке ansible запустить ansible-playbook с следующими параметрами:

```
ansible-playbook -i environment/prod/inventory.yaml \
                playbooks/deploy_pgcluster.yml\
                -u %USER_NAME%
        ansible-playbook -i environment/prod/inventory.yaml \
                playbooks/configure_pg_cluster.yaml\
                -u %USER_NAME%
        ansible-playbook -i environment/prod/inventory.yaml \
                playbooks/monitoring.yaml \
                -u %USER_NAME%
```

Значение %USER_NAME% - имя пользователя, с привилегиями администратора, от которого производятся операции в системе виртуальной машины.

Контакты технических специалистов, которые могут проконсультировать по процессу развёртывания и настройки экземпляра ПО и его функционирования: Аблязов Марат, m.ablyazov@crpt.ru, тел.: 89161286727.