ИНСТРУКЦИЯ ПО УСТАНОВКЕ

СИСТЕМЫ ТЕХНИЧЕСКОГО ДОКУМЕНТООБОРОТА

ФЕНИКС

Автор:	ООО «Цифровая Эпоха»
Дата создания:	14.07.2023
Дата обновления:	16.09.2023
Версия:	1.4

Санкт-Петербург, 2023

ОГЛАВЛЕНИЕ

1 Инструкция по установке ПО Феникс

1.1 Системные требования

Для развертывания кластера Kubernetes и компонентов ПО Феникс требуется:

Роль сервера	vCPU	RAM	Объем дискового пространства (ГБ)	ОС
Мастер нода	6	10	100	Debian11
Воркер нода	6	12	100	
Воркер нода	6	12	100	
Воркер нода	6	12	100	

- Зарезервированный пул адресов (32) для virtualIP в кластере kubernetes

1.2 Требования к АРМ

Требования к вычислительным ресурсам рабочей станции конечного пользователя:

- Intel core 2 Quad 2.4 GHz
- 6GB RAM
- 10GB выделенного пространства на диске для работы с файлами
- Доступ к сети окружения Системы
- Windows 10 (разрядность ОС 64 бит)
- Браузер на базе Chromium (не ниже версии 116.0.5845.142) – Yandex Browser

1.3 Требования к специалистам, выполняющим установку

Для выполнения работ по установке программного обеспечения Феникс специалист должен обладать следующими знаниями:

- архитектуру, состав, функции и команды операционной системы Debian, принципы установки и настройки основных компонентов операционной системы;
- протокол сетевого взаимодействия TCP/IP;
- основы баз данных PostgreSQL, основы языка SQL;
- общую структуру баз данных первого и второго уровня;
- назначение основных таблиц БД и взаимосвязь данных;

- состав прикладного программного обеспечения ФЕНИКС, порядок установки и настройки ПО; конфигурационные файлы, назначение разделов и параметров;
- принципы контейнеризации, принципы работы оркестраторов, в частности Kubernetes (инструменты управления пл. Kubernetes).

1.4 ПО Феникс. Общие технические сведения

Система ФЕНИКС представляет собой набор микросервисных приложений, взаимодействие которых осуществляется по общей шине. Микросервисы платформы ФЕНИКС разворачиваются, запускаются и управляются в Kubernetes.

Основные компоненты Системы:

- Kubernets-кластер, состоящий из нескольких узлов (серверов) под управлением ОС Debian11/Astra Linux 1.7 (для Deckhouse)
- Инфраструктурных компонент (СУБД, RabbitMQ)
- Вспомогательных компонент
- Микросервисные приложения (поды)
- Сервисы Kubernetes
- Балансировщик нагрузки.

Система может быть интегрирована:

- С Microsoft Active Directory по LDAP
- С почтовым сервером по SMTP и IMAP.

1.5 Подготовительные работы

Перед производством работ по установке и настройке ПО ФЕНИКС необходимо подготовить:

1. Количество N ВМ/серверов под управлением Debian 11. Количество серверов (N) напрямую зависит от количества пользователей и документов.
2. Доступ к оболочке ОС
3. Устройства ввода (клавиатура, мышь)
4. Устройство вывода (монитор)
5. Предварительно скачанный набор дистрибутивов и образов ФЕНИКС, либо доступ к репозиториям ФЕНИКС и соответствующему версии ОС репозиторию Debian

6. Обеспечить сетевую доступность между серверами кластера и АРМ специалиста, выполняющего установку
7. Установить на АРМ специалиста, выполняющего установку браузер на базе Chromium (не ниже версии 116.0.5845.142) – Yandex Browser.

Целевым серверам назначается соответствующая кластеру Kubernetes роль: мастер или воркер.

Убедившись, что сервера доступны по локальной сети, на сервере с ролью «мастер» необходимо подключиться к оболочке ОС по ssh, либо напрямую.

2 Установка программного обеспечения

2.1 Предварительная установка пакетов

Все команды выполняются под пользователем root.

Для установки утилит и платформы Kubernetes необходимо последовательно выполнить команды на всех серверах кластера:

1. Установить python, helm и редактор nano:
   ```
   apt install python -y
   ```
   ```
   apt install helm -y
   apt install nano -y
   ```
2. Отключить swap:
   ```
   swapoff -a
   ```
3. Создать пользователя администратора kubernetes:
   ```
   useradd -m kuser
   usermod -aG sudo kuser
   usermod -a -G root kuser
   passwd kuser
   ```
   (ввести пароль два раза)
4. Изменить rc.local:
   ```
   sudo nano /etc/rc.local
   ```
   ```
   #!/bin/bash
   ```
   ```
   dhclient
   ```
   ```
   exit 0
   ```

```

```
sudo chmod 755 /etc/rc.local

sudo systemctl enable rc-local

sudo systemctl restart rc-local

sudo systemctl status rc-local
```

5. Создать каталог /opt/k8s:

```
mkdir /opt/k8s
chown kuser /opt/k8s
```

6. Установить python3 и ansible:

```
apt install python3 python3-pip
apt-get install ansible
```

7. Войти в оболочку ОС сервера с ролью «мастер» под пользователем kuser

8. Сгенерировать ssh-ключ:

```
ssh-keygen
"<пароль>"
location ngcp-dev-mst01: /home/kuser/.ssh/id_rsa
```

9. Скопировать ключ на остальные сервера кластера:

```
ssh-copy-id -p 22 kuser@<имя сервера>
```

10. Установить kubespray:

```
sudo apt install git
sudo apt-get install python3-venv
curl "https://bootstrap.pypa.io/get-pip.py" -o "get-pip.py"
python3 get-pip.py
cd /opt/k8s
git clone https://repo.digitalepoch.ru/kubernetes-sigs/kubespray.git &&
cd kubespray
git checkout v2.21.0
cp -rfp inventory/sample inventory/ngcp
```

11. Отредактировать содержимое файла /opt/k8s/inventory/ngcp/inventory.ini:

```
[all]
<имя мастера> ansible_host=<IPv4-адрес мастера> ansible_port=22
<имя воркера> ansible_host=<IPv4-адрес воркера> ansible_port=22
[kube_control_plane]
<имя мастера>
```
```

```
[etcd]
<имя мастера>
[kube_node]
<имена воркеров через отступ>
[calico_rr]
[k8s_cluster:children]
kube_control_plane
kube_node
calico_rr
```

12. Отредактировать содержимое файла /opt/k8s/inventory/ngcp/group_vars/k8s_cluster/k8s-cluster.yml:

```
kube_proxy_strict_arp: true
```

13. Отредактировать содержимое файла /opt/k8s/inventory/ngcp/group_vars/k8s_cluster/addons.yml:

```
metrics_server_enabled: true
metrics_server_kubelet_insecure_tls: true
metrics_server_metric_resolution: 15s
metrics_server_kubelet_preferred_address_types: "InternalIP"
ingress_nginx_enabled: true
ingress_nginx_host_network: false
ingress_publish_status_address: ""
ingress_nginx_nodeselector:
kubernetes.io/os: "linux"
ingress_nginx_namespace: "ingress-nginx"
ingress_nginx_insecure_port: 80
ingress_nginx_secure_port: 443
ingress_nginx_configmap:
map-hash-bucket-size: "128"
ssl-protocols: "TLSv1.2 TLSv1.3"
metallb_enabled: true
metallb_speaker_enabled: true
metallb_ip_range:
   - "<начальный IPv4-адрес пула>-<конечный IPv4-адрес пула>"
```

14. Установка requirments:

```
mkdir venv
python3 -m venv ./venv
```

```
source ./venv/bin/activate
pip install -r requirements.txt
```

15. Запуск ansible:

```
ansible-playbook -u kuser -i /opt/k8s/kubespray/inventory/ngcp
/inventory.ini cluster.yml -b --diff --extra-vars
"ansible_sudo_pass=<пароль пользователя kuser>"
```

2.2 Установка инфраструктурных и вспомогательных компонент

Далее описан процесс установки инфраструктурных вспомогательных сервисов, инфраструктурных компонентов и вспомогательных сервисов. Все действия выполняются под пользователем kuser на сервере с ролью «мастер».
Для доступа к репозиторию необходимо заранее получить УЗ. УЗ для доступа к docker-репозиторию Феникс после получения необходимо прописать в values.yaml (п.3).

На сервере с ролью «мастер» необходимо последовательно выполнить операции:

1. Распаковать содержимое ngcp-manifests.tar в `/opt/k8s/kubespray/inventory/ngcp/microservices`
2. Перейти в директорию:
   `cd /opt/k8s/kubespray/inventory/ngcp/microservices`
3. Добавить файл с переменными values.yaml. *Заполнить значения переменных <>:*


values.yaml

4. Авторизоваться в repo. Для авторизации в терминале требуется ввести логин/пароль:
   `sudo helm repo add ngcp-helm https://repo.digitalepoch.ru/ --username <имя_пользователя_репозитория> --password <пароль_пользователя_репозитория>`
5. Для проверки подключения к репозиторию необходимо выполнить команды:

```
helm repo update
helm show all ngcp-helm/ngcp-front --version 0.1.0
```

6. Установить инфраструктурные компоненты (БД, RabbitMQ)
   `sudo helm install ngcp-infra -f values.yaml --version 0.1.0 ngcp-helm/infra-charts --create-namespace --namespace ngcp`
7. *Установка сервисов занимает некоторое время, для перехода к следующему шагу необходимо убедиться, что сервисы запустились корректно и*

отсутствуют ошибки. Выполнить команду:
`sudo kubectl get pods -n ngcp`

8. Установить вспомогательные сервисы (KeyCloak, OnlyOffice):
`sudo helm install ngcp-sup-services -f values.yaml --version 0.1.0 ngcp-helm/ngcp-sup-services --create-namespace --namespace ngcp`

9. *Установка сервисов занимает некоторое время, для перехода к следующему шагу необходимо убедиться, что сервисы запустились корректно и отсутствуют ошибки. Выполнить команду:*
`sudo kubectl get pods -n ngcp`

2.3 Установка прикладных сервисов ФЕНИКС

Далее описан процесс установки прикладных сервисов, инфраструктурных компонентов и вспомогательных сервисов. Все действия выполняются под пользователем kuser на сервере с ролью «мастер».

1. Перейти в директорию:
`cd /opt/k8s/kubespray/inventory/ngcp/microservices`

2. Для установки сервисов из удаленного registry выполнить следующую команду:
`sudo helm install ngcp-services -f values.yaml --version 0.1.0 ngcp-helm/ngcp-chart --create-namespace --namespace ngcp`

3. *Установка сервисов занимает некоторое время, для перехода к следующему шагу необходимо убедиться, что сервисы запустились корректно и отсутствуют ошибки. Выполнить команду:*
`sudo kubectl get pods -n ngcp`

2.4 Установка прикладных фронтенд-сервисов ФЕНИКС

1. Перейти в директорию:
`cd /opt/k8s/kubespray/inventory/ngcp/microservices`

2. Для установки сервисов из удаленного registry выполнить следующую команду:
`sudo helm install ngcp-front -f values.yaml --version 0.1.0 ngcp-helm/ngcp-front --create-namespace --namespace ngcp`

3. *Установка сервисов занимает некоторое время, для перехода к следующему шагу необходимо убедиться, что сервисы запустились корректно и отсутствуют ошибки. Выполнить команду:*
`sudo kubectl get pods -n ngcp`

2.5 Проверка доступности веб-приложения ФЕНИКС

Для проверки доступности веб-приложения необходимо на АРМ специалиста, выполнявшего установку пройти по адресу: https://<адрес балансировщика>:443/. Должно открыться окно веб-приложение (фронтенд) ПО Феникс (рисунок 1).

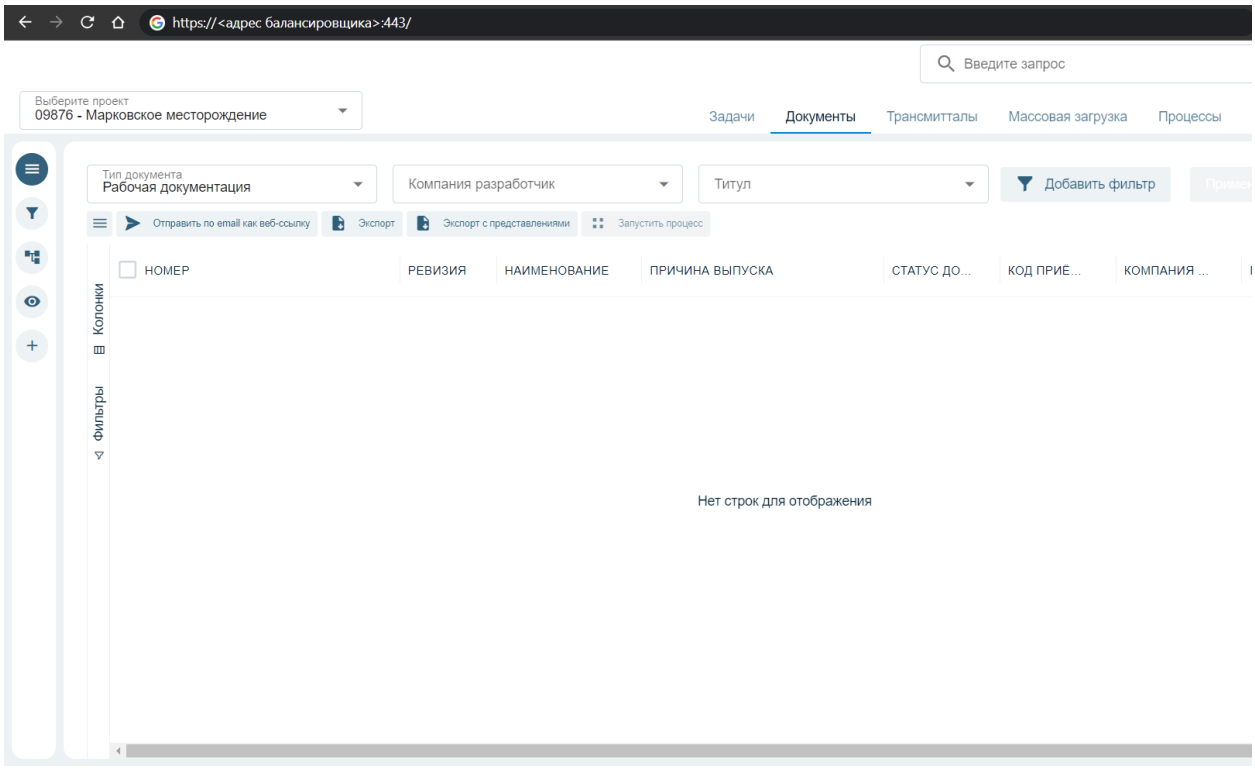

Рисунок 1 – Окно программы ФЕНИКС.

3 Администрирование ПО Феникс

3.1 Управление кластером Kubernetes

Все операции по управлению кластером Kubernetes выполняются на сервере с ролью «мастер» под пользователем kuser.

- Для отображения списка всех подов и их статуса необходимо выполнить команду:
  sudo kubectl get pods -o wide --all-namespaces
- Для отображения списка всех сервисов и их статусов необходимо выполнить команду:
  sudo kubectl get svc -o wide --all-namespaces
- Для просмотра информации по запущенному поду необходимо выполнить команду:
  sudo kubectl describe pod <имя пода> --namespace=ngcp
- Для просмотра по сервису необходимо выполнить команду:
  sudo kubectl describe svc <имя сервиса> --namespace=ngcp
- Для изменения конфигурации пода напрямую необходимо выполнить команду:
  sudo kubectl edit pod <имя пода> --namespace=ngcp
- Для изменения конфигурации сервиса напрямую необходимо выполнить команду:
  sudo kubectl edit svc <имя сервиса> --namespace=ngcp
- Для изменения параметров metallb-system необходимо выполнить команды:
  sudo kubectl edit configmap config -n metallb-system
  sudo kubectl apply -f configmap config -n metallb-system

Поставляемые приложения уже имеют предустановленные параметры, как на уровне приложения, так и на уровне манифестов. Вышеописанные команды требуется выполнять только в случае крайней необходимости изменения инфраструктурных параметров.

Для более подробного ознакомления с механизмами управления и командами kubectl рекомендуется изучить официальную документацию: https://kubernetes.io/ru/docs.

3.2 Администрирование PostgreSQL

Для администрирования БД PostgreSQL рекомендуется использовать платформу pgAdmin4. Для установки и подключения рекомендуется:

1. Скачать дистрибутив: https://www.pgadmin.org/download/
2. Установить pgAdmin4 по инструкции, отраженной на странице загрузки (п.1)

3. Запустить pgAdmin4
4. Настроить подключение указав: адрес сервиса postgresql, порт, пользователя БД.

Для получения более подробной информации о возможностях pgAdmin4 рекомендуется ознакомиться с документацией, доступной на сайте: https://www.pgadmin.org/docs/pgadmin4/7.6/index.html; а также с документацией PostgreSQL

3.3 Настройка клиента KeyCloak

Для доступа к Keycloak со стороны API шлюза, API шлюз требуется зарегистрировать в Keycloak. Для этого требуется:

- Войти в интерфейс администрирования.
- Выбрать нужный Realm.
- Перейти в раздел Clients.
- Нажать кнопку Create client.
- В разделе General Settings заполнить поля:
- Client type - OpenID Connect;
- Client ID - идентификатор клиента, например, ngcp-gw-api-client.
- В разделе Capability Config включить галочку Client Authentication.
- В разделе Login Settings заполнить поля:
- Root URL - базовый адрес API шлюза, например, https://api-gw.ngcp.local;
- Home URL - домашняя страница, например, /index
- Valid redirect URIs - допустимые адреса перенаправления для передачи кода аутентификации, например, /login/oauth2/code/.
- Valid post logout redirect URIs - допустимые адреса перенаправления, после выполнения Logout.

После создания клиента, пароль можно получить с вкладки Credentials в окне настроек клиента.

3.4 Настройка Keycloak Scopes

Spring Security в процессе аутентификации производит запрос к ресурсу UserInfo. Keycloak запрещает доступ к этому ресурсу для клиентов, у которых не задан Scope

openid. Изначально, такой Scope отсутствует в списке доступных в Keycloak. Требуется создать его вручную. Для этого нужно:

- Войти в интерфейс администрирования.
- Выбрать нужный Realm.
- Перейти в раздел Client scopes.
- Нажать кнопку Create client scope.
- Заполнить поле Name = openid и сохранить Scope.
- Далее, нужно назначить клиенту Scope openid. Для этого нужно:
- Перейти в раздел Clients.
- Открыть на редактирование нужный клиент.
- Перейти на вкладку Client scopes.
- Назначить Scope openid, при помощи кнопки Add client scope.