Протокол № 24-002 проведения совместных испытаний “Deckhouse Kubernetes Platform” версии 1.59.12 и 1.60.4 и системы виртуализации “РЕД Виртуализация” 7.3.

г. Москва 03.06.2024

## Предмет испытаний

В настоящем протоколе зафиксирован факт проведения в период с 10.06.2024 по 13.06.2024 совместных испытаний программного обеспечения «Deckhouse Kubernetes Platform» версии 1.59.12 и 1.60.4 (далее – ПО), разработанного АО «Флант», и системы виртуализации “РЕД Виртуализация” версии 7.3 (далее - Платформа виртуализации), разработанной ООО “РЕД СОФТ”.

## Объект испытаний

Перечень компонентов, эксплуатировавшихся в ходе проведения данных испытаний, относящихся к ПО, приведен в Таблице 1.

*Таблица 1 - Перечень компонентов, относящихся к ПО*

| Описание | Наименование | Источник |
|---|---|---|
| Файл программного пакета дистрибутива ПО | Deckhouse EE v 1.59.12 и 1.60.4 | Источник в сети “Интернет”, адрес: https://registry.deckhouse.io/dechouse/ee/ |
| Официальное руководство по эксплуатации ПО в электронном формате | Deckhouse Platform на bare metal | https://deckhouse.ru/gs/bm/step2.html |

## Ход испытаний

1. В ходе выполнения настоящих испытаний были выполнены проверки корректности Функционала ПО на Платформе виртуализации в объёме, указанном в Приложении 1.

2. Перечень официальных репозиториев ПО, которые эксплуатировались в ходе испытаний:
    a. Платформа виртуализации “РЕД Виртуализация” 7.3.
    b. “Deckhouse Kubernetes Platform” 1.59.12 и 1.60.4
    c. РЕД ОС 7.3 Муром
3. Неофициальные репозитории ПО не использовались в рамках испытаний.
4. Установка ПО производится с отдельного установочного узла на целевой узел под управлением тестируемой ОС, данная информация отражена в Приложении 2.
5. В ходе испытаний рассматривался вариант установки ПО в статическом кластере при при использовании функционала Deckhouse для настройки статических узлов кластера, без интеграции с API Платформы виртуализации.

## Результат испытаний

ПО корректно функционирует в кластере, состоящем из статических ВМ, которые развернуты вручную средствами Платформы виртуализации.

## Вывод

1. ПО и Платформа виртуализации совместимы, учитывая п. 4 “Ход испытаний”, раздел “Результат испытаний” и Приложении 2.

## Состав рабочей группы и подписи сторон

Данный протокол составлен участниками рабочей группы:

- Со стороны ПО (АО “Флант”):
    - Архитектор инфраструктурных решений Салеев К.Ю.
    - Девопс-инженер Головачев С.С.
- Со стороны ООО “РЕД СОФТ”
    -

# Приложение 1 к Протоколу № 24-002

**Перечень проверок совместимости ПО и Платформы виртуализации**

*Таблица 2 - Список проверок*

| № п/п | Наименование проверки | Результат проверки |
|---|---|---|
| 1 | Установка ПО на master-узел | Успешно |
| 2 | Установка ПО на worker-узел | Успешно |
| 3 | Эксплуатация минимальной базовой версии ПО | Успешно |
| 4 | Запуск, остановка выполнения ПО | Успешно |
| 5 | Остановка ПО | Успешно |
| 6 | Восстановление работы ПО после перезапуска ВМ | Успешно |

Приложение 2 к Протоколу № 24-002

**Инструкция по выполнению проверки совместимости ПО и Платформы виртуализации**

*Таблица 3 - Инструкции по проверке совместимости*

| № | Шаг | Описание шага | Ожидаемый результат | Примечания |
|---|---|---|---|---|
| ***Этап 1: Подготовительный*** | | | | |
| 1.1 | Получить лицензионный ключ для Deckhouse EE | Получить от коммерческого департамента «Флант» ключ для EE версии | Ключ получен | |
| 1.2 | Выбрать инфраструктуру для установки типа bare metal | https://deckhouse.ru/gs/index.html | Выбран тип установки Bare metal | Выбрать пункт Bare metal |
| 1.3 | Подготовить ПК согласно требованиям | На странице https://deckhouse.ru/gs/bm/step2.html перейти к пункту "Что необходимо для установки" и раскрыть пункт "Требования" в разделе "Персональный компьютер" | ПК для установки соответствует требованиям | |

| | | | | |
|---|---|---|---|---|
| 1.4 | Подготовка ВМ для master-ноды | На странице https://deckhouse.ru/gs/bm/step2.html перейти к пункту "Что необходимо для установки" и раскрыть пункт "Требования" в разделе "Физический сервер или виртуальная машина для master-узла" | Создан master-узел с необходимими ТТХ | При настройке узла должны быть учтены следущие пункты:<br><br>- Добавлен пользователь<br>- Добавлена публичная часть ssh-ключа, с которым будет осуществлён доступ на ВМ<br>- При настройке тайм-зоны необходимо указать корректную тайм-зону (чтобы избежать некорректной работы в дальнейшем, например UTC +0300)<br>- ВМ должна быть минимум 4 vCPU, 8 RAM, 50 Gb SSD |
| 1.5 | Указать шаблон DNS-имен кластера | Выполнить по инструкции на странице https://deckhouse.ru/gs/bm/step3.html | Задан шаблон DNS | |
| 1.6 | Формирование файла config.yml для установки | - На шаге https://deckhouse.ru/gs/bm/step4.html произвести настройку config.yml (какие необходимы параметры)<br>- Обратить вниманием на параметр internalNetworkCIDRs (список внутренних сетей узлов кластера): внутренние сети используются для связи компонентов Kubernetes (используется при наличии более одной подсети, для организации связности узлов кластера) | config.yml сформирован и отредактирован под требования к установке | Секция general<br><br>- При формировании config.yml нужно будет добавить в настройки genaral localpath для worker-узлов. storageClass: localpath-deckhouse-worker<br>- Т.к. кластер может быть без выхода в интернет, то для простоты инсталяции нужно узкать, что в кластере будут самоподписные сертификаты:<br><br>....<br>spec:<br>version: 1<br>settings:<br>modules:<br>https:<br>mode: CertManager<br>certManager:<br>clusterIssuerName: selfsigned |
| ***Этап 2: Первичная установка*** | | | | |

| | | | | |
|---|---|---|---|---|
| 2.1 | Запустить установщик (docker container) | Выполнить команду с шага "Запустите установщик на персональном компьютере." на странице https://deckhouse.ru/gs/bm/step4.html | - Успешная авторизация в registry.deckhouse.io<br>- Выкачивание образа установщика deckhouse<br>- Запуск docker-контейнера | Примечаение-1: Запуск установщика может быть выполнен как с локального ПК, так и с иных машин (например, bastion)<br><br>Примечание-2: Пароль замаскирован в виде **********, ваш пароль будет сформирован под ваш лицензионный ключ.<br><br>base64 -d <<< ********** \| docker login -u license-token --password-stdin registry.deckhouse.ru<br>docker run --pull=always -it -v "$PWD/config.yml:/config.yml" -v "$HOME/.ssh/:/tmp/.ssh/" registry.deckhouse.ru/deckhouse/ee/install:stable bash |
| 2.2 | Запуск установки master-узла | Выполнить команду на установку master-узла | - Запустился bootstrap-скрипт для master-узла<br>- Через некоторое время (зависит от ВМ), скрипт должен завершиться успешно с сообщением:<br><br>Финальное сообщение: "🎉 Deckhouse cluster was created successfully!" | Пример команды (для подключения на мастер-узел с локального ПК):<br>dhctl bootstrap --ssh-user=<username><br>--ssh-host=<master_ip><br>--ssh-agent-private-keys=/tmp/.ssh/id_rsa \<br>--config=/config.yml \<br>--ask-become-pass |

| | | | | |
|---|---|---|---|---|
| 2.3 | Установить LocalPathProvisioner | Выполняется на master-узле<br><br>- Выполнить команду<br><br>kubectl create -f - << EOF<br>apiVersion:<br>deckhouse.io/v1alpha1<br>kind: LocalPathProvisioner<br>metadata:<br>name:<br>localpath-deckhouse-worker<br>spec:<br>nodeGroups:<br>- worker<br>path:<br>"/opt/local-path-provisioner"<br>EOF | - Команда отработала<br>- Создался путь<br><br>localpathprovisioner.deckhouse.io/localpath-deckhouse-worker created | |
| ***Этап 3: Добавление worker-узла при помощи Cluster API Provider Static*** | | | | |
| 3.1 | Предварительная подготовка | Подготовьте необходимые ресурсы — серверы/виртуальные машины, установите специфические пакеты ОС, добавьте точки монтирования, настройте сетевую связанность и т.п. | Машины подготовлены по требованиям. | Подготовить 1 узел для worker (4vCPU, 16 RAM, 100Gb SSD) |

| | | | | |
|---|---|---|---|---|
| 3.2 | Подготовить данные по SSH ключам для доступа на ВМ | - Подготовить приватную часть SSH ключа в формате PEM, закодированный в base64 на мастер-узле<br><br>-- ssh-keygen -t rsa -f caps-id -C "" -N ""<br>-- Для примера: создалось 2 файла: caps-id и caps-id.pub<br>-- Для шифрования: base64 -w0 caps-id (выполняется в директории .ssh)<br>-- На выходе получится строка в формате base64, которую необходимо использовать далее<br><br>- Подготовить username для доступа на ВМ | - Ключ подготовлен<br>- Пользователь готов, обладает правами sudo | |
| 3.3 | Добавить на отдельный узел ssh ключ | - Добавить на ВМ (созданная для worker-узла) ssh-ключ в authorized_keys<br>- Ключ должен быть добавлен под тем пользователем, с которым будет производиться подключение к ВМ (подробнее про пользователя на шаге 3.5)<br><br>echo 'значение_casp-id.pub' >> ~/.ssh/authorized_keys | Публичная часть ssh-ключа добавлена | |

| | | | | |
|---|---|---|---|---|
| 3.4 | Подготовительный шаг на master-узле | **Выполняется на master-узле**<br><br>- Перейти в sudo<br>- Создать перерменные окружения со значанием приватного ключа в формате base64<br>(чтобы объявить переменную для использования в скриптах на создание SSHCredentials) | Переменная создана | PRIVATE_KEY_BASE64=$(cat ~/.ssh/caps-id \| base64 -w0)<br><br>echo $PRIVATE_KEY_BASE64 |
| 3.5 | Создать ресурс SSHCredentials | **Выполняется на master-узле**<br><br>- Создать ресурс по документации (https://deckhouse.ru/documentation/v1/modules/040-node-manager/cr.html#sshcredentials) | Ресурс создан<br><br>sshcredentials.deckhouse.io/credentials created | #Создание ресурса SSHCredentials<br>kubectl apply -f - <<EOF<br>apiVersion: deckhouse.io/v1alpha1<br>kind: SSHCredentials<br>metadata:<br>name: credentials<br>spec:<br>user: username<br>privateSSHKey: ${PRIVATE_KEY_BASE64}<br>sshPort: 22<br>EOF |

| | | | | |
|---|---|---|---|---|
| 3.6 | Создать ресурс StaticInstance | **Выполняется на master-узле**<br><br>- Создать ресурс по документации (https://deckhouse.ru/documentation/v1/modules/040-node-manager/cr.html#staticinstance) | Ресурс создан<br><br>staticinstance.deckhouse.io/worker-0 created | # Добавление worker-инстанса<br>kubectl apply -f - <<EOF<br>apiVersion: deckhouse.io/v1alpha1<br>kind: StaticInstance<br>metadata:<br>name: worker-0<br>labels:<br>role: worker<br>spec:<br>address: WORKER_IP #Указать IP-адрес машины<br>credentialsRef:<br>kind: SSHCredentials<br>name: credentials<br>EOF |
| 3.7 | Создать NodeGroup | **Выполняется на master-узле**<br><br>- Создать ресурс по документации (https://deckhouse.ru/documentation/v1/modules/040-node-manager/cr.html#nodegroup) | Ресурс создан<br><br>nodegroup.deckhouse.io/worker created | # Создание NodeGroup worker<br>kubectl apply -f - <<EOF<br>apiVersion: deckhouse.io/v1<br>kind: NodeGroup<br>metadata:<br>name: worker<br>spec:<br>nodeType: Static<br>staticInstances:<br>count: 1<br>labelSelector:<br>matchLabels:<br>role: worker<br>EOF |

| | | | | |
|---|---|---|---|---|
| 3.8 | Проверка состояния кластера после установки | Выполняется на master-узле<br><br>- Выполнить команду kubectl get kubectl get staticinstances.deckhouse.io<br>- Убедиться, что интансы в статусе Running<br>- Выполнить команду kubectl get ng<br>- Убедиться, что node groups созданы<br>- Выполнить команду kubectl get no<br>- Убедиться, что worker-узел добавлен в кластер и находится в состоянии Ready | - Инстансы в Ready<br>- NodeGroups созданы<br>- worker-узел в статусе Ready | |
| ***Этап 4: Проверка состояния кластера*** | | | | |
| 4.1 | Итоговая проверка | Проверки:<br><br>- kubectl get no - проверить, что API k8s работает и отображает список узлов<br>- kubectl get po -A - вывести список всех подов во всех namespaces | - Отображаются списки узлов<br>- Отображается список подов, убедиться что нет подов в состоянии отличном от Running | |