Протокол № 24-004 проведения совместных испытаний “Deckhouse” версии 1.60.6 и Платформы виртуализации zVirt версии 4.0.

г. Москва 01.07.2024

Предмет испытаний

В настоящем протоколе зафиксирован факт проведения в период с 18.06.2024 по 24.06.2024 совместных испытаний программного обеспечения «Deckhouse» версии 1.60.6 (далее – ПО), разработанного АО «Флант», и системы виртуализации zVirt версии 4.0 (далее - Платформа виртуализации), разработанной ООО “Орион”.

Объект испытаний

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

Таблица 1 - Перечень компонентов, относящихся к ПО

Описание	Наименование	Источник
Файл программного пакета дистрибутива ПО	Deckhouse EE v 1.60.6	Источник в сети “Интернет”, адрес: https://registry.deckhouse.ru/dechouse/ee/install:v1.60.6
Официальное руководство по эксплуатации ПО в электронном формате	Deckhouse Platform на zVirt	https://deckhouse.ru/gs/zvirt/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 zVirt 4.0.
    b. Deckhouse Kubernetes Platform 1.60.6
    c. ALT Server 10.1 (Mendelevium)
3. Неофициальные репозитории ПО не использовались в рамках испытаний.
4. Установка ПО производится с отдельного установочного узла (bastion-host).
5. Испытания включали в себя сценарии, использующие интеграцию с API платформы виртуализации.

Результат испытаний

ПО корректно функционирует на ресурсах в кластере, которые были развёрнуты при интеграции с API Платформы виртуализации. В ходе функциональных проверок подтверждена корректная работа с API Платформы виртуализации.

Вывод

1. ПО и Платформа виртуализации совместимы, учитывая п 4 “Ход испытаний”, раздел “Результат испытаний” и Приложение 2.
2. ПО имеет ограничение по работе с API Платформы виртуализации в части работы с дисками (нет возможности управлять размером диска из ПО, только через консоль/интерфейс Платформы виртуализации).

Состав рабочей группы и подписи сторон

Данный протокол составлен участниками рабочей группы:

- Со стороны ПО (АО “Флант”):
    - Архитектор инфраструктурных решений Салеев К.Ю.
    - DevOps-инженер Лазарев Е.В.

Приложение 1 к Протоколу № 24-004

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
7	Автоматическое обновление платформы Deckhouse	Успешно
8	Поддержка РФ операционных систем	Успешно
9	Обновление версии Kubernetes	Успешно
10	Возможность увеличения количества control-plane узлов	Успешно
11	Управление узлами кластера	Успешно
12	Автоматическая настройка узлов кластера	Успешно
13	Возможность дополнительной конфигурации runtime-компонентов узлов кластера	Успешно
14	Размещение компонентов Deckhouse Kubernetes Platform на выделенных узлах	Успешно
15	Запуск модулей Deckhouse Enterprise версии	Успешно
16	Установка / добавление элементов интерфейса / модулей (из поставки платформы)	Успешно
17	Возможность отключения неиспользуемых модулей	Успешно

	платформы	
18	Отказоустойчивая конфигурация всех компонентов платформы	Успешно
19	Управление namespaces (добавление, удаление, редактирование)	Успешно
20	Возможность использования внешних модулей	Успешно
21	Аудит событий Kubernetes API	Успешно
22	Фильтрации трафика внутри кластера	Успешно
23	Фильтрации трафика на уровне L7 внутри кластера	Успешно
24	Отображения действия политик (NetworkPolicy) в веб-интерфейсе	Успешно
25	Возможность использования корпоративного TLS/SSL сертификата для компонентов платформы	Успешно
26	Использования временных статических пользователей в кластере	Успешно
27	Использование статических групп пользователей в кластере	Успешно
28	Использование внешнего провайдера аутентификации (LDAP/AD/OIDC)	Успешно
29	Настройка ролевой модели доступа на основе групп, атрибутов пользователя	Успешно
30	Ограничение доступа пользователей к определенным namespace	Успешно
31	Возможность расширения прав доступа	Успешно
32	Использование сервисной учетной записи для выката прикладного ПО в платформу	Успешно
33	Использование политик безопасности Kubernetes (Pod Security Standards)	Успешно
34	Использование операционных политик для безопасной работы прикладного ПО	Успешно
35	Использование политик безопасности для безопасной работы прикладного ПО	Успешно

36	Возможность использовать квот в рамках namespaces	Успешно
37	Создание изолированного окружения по заготовленному шаблону	Успешно
38	Обнаружение угроз безопасности анализирую прикладное ПО и контейнеры	Успешно
39	Организация mTLS между узлами прикладного ПО	Успешно
40	Организация авторизации доступа между сервисами	Успешно
41	Сканирование образов прикладного ПО на наличие известных уязвимостей	Успешно
42	Встроенный мониторинг состояния служебных компонент кластера	Успешно
43	Мониторинг аппаратных ресурсов платформы	Успешно
44	Мониторинг Kubernetes в составе платформы	Успешно
45	Встроенный мониторинг входящего трафика	Успешно
46	Оценка использования ресурсов	Успешно
47	Уведомления (alerts) по нагрузке серверов кластера, количество ошибочных запросов ingress и пр.	Успешно
48	Расширенный мониторинг состояния прикладных сервисов	Успешно
49	Мониторинг прикладных сервисов	Успешно
50	Возможность добавления своего набора уведомлений (alerts)	Успешно
51	Возможность отправки уведомлений (alerts) во внешнюю систему	Успешно
52	Балансировка нагрузки контейнеров между узлами кластера	Успешно
53	Масштабирование прикладных сервисов на основе бизнес метрик	Успешно
54	Масштабирование прикладных сервисов на основе	Успешно

	потребления ресурсов	
55	Автоматические масштабирование количества узлов кластера	Успешно
56	Автоматические распределение ресурсов между узлами кластера	Успешно
57	Встроенная возможность автоматического распространения secrets	Успешно
58	Автоматический перезапуск прикладного ПО в случае изменения secret / configmap	Успешно
59	Настройка входящего трафика для кластера (Ingress)	Успешно
60	Встроенные инструменты удаленного ведения и агрегации журналов (логов)	Успешно
61	Встроенная система кратковременного хранения логов	Успешно
62	Доступ к кластеру через OpenVPN	Успешно

Приложение 2 к Протоколу № 24-004

Инструкция по выполнению проверки совместимости ПО и Платформы виртуализации

Таблица 3 - Инструкции по проверке совместимости

№ п/п	Описание требования	Критерий оценки успешности проверки	Раздел документации
1	Возможность установки платформы в закрытом контуре	Произвести установку Deckhouse Platform в закрытом окружении (без доступа в интернет) - установка прошла успешно	Deckhouse Platform в закрытом окружении
2	Автоматическое обновление платформы Deckhouse	Установить платформу версии на единицу меньше в миноре, чем на выбранном канале обновления (https://releases.deckhouse.ru/) с автоматическим механизмом обновления. Убедиться, что Deckhouse обновился и deckhouserelease выбранного канала обновления находится в статусе Deployed	Обновление Deckhouse
3	Возможность ручного обновления платформы Deckhouse	Установить платформу версии на единицу меньше в миноре, чем на выбранном канале обновления (https://releases.deckhouse.ru/) с ручным механизмом обновления, произвести обновление. Убедиться, что Deckhouse обновился и deckhouserelease выбранного	Обновление Deckhouse

		канала обновления находится в статусе Deployed	
4	Поддержка РФ операционных систем (РЕДОС, ALT linux, Astra Linux)	Установить Deckhouse Platform на узел под управлением РФ ОС и убедиться, что поды запускаются успешно	Поддерживаемые версии ОС
5	Обновление версии Kubernetes	Произвести первоначальную установку кластера с версией Kubernetes “Automatic”. Изменить версию Kubernetes на желаемую и убедиться, что на узлах кластера kubelet необходимой версии	Как обновить версию Kubernetes в кластере
6	Возможность увеличения количества control-plane узлов	Произвести добавление еще двух master узлов - убедиться в работоспособности кластера	Как добавить master-узел в статичном или гибридном кластере
7	Управление узлами кластера (добавление, удаление)	Добавить узел к Kubernetes кластеру и убедиться, что узел успешно добавлен в кластер и перешел в статус Ready. Затем освободить узел от рабочих нагрузок и удалить из кластера. Убедиться, что узла нет в кластере.	Как добавить статичный узел в кластер
8	Автоматическая настройка узлов кластера	Применить ресурс NodeGroupConfiguration и проверить работу systemd сервиса bashbile на узле кластера - убедиться, что параметр на узле кластера изменился на необходимое значение	Пример задания параметра sysctl

9	Возможность дополнительной конфигурации runtime-компонентов узлов кластера	Применить настройку ресурса NodeGroup и проверить работу systemd сервиса bashbile на узле кластера - убедиться, что параметр на узле кластера изменился на необходимое значение	Изменение параметров runtime-компонентов
10	Размещение компонентов Deckhouse Kubernetes Platform на выделенных узлах	Установить выделенный узел под monitoring компоненты - убедиться, что prometheus запустился на данных узлах кластера Kubernetes	Управление размещением компонентов Deckhouse
11	Запуск модулей Deckhouse Enterprise версии	Включить модуль user-authn и убедиться, что создался namespace d8-user-authn и в нем есть под dex	Включение и отключение модуля
12	Установка / добавление элементов интерфейса / модулей (из поставки платформы)	- По умолчанию развертывается набор модулей default - После установки возможно изменение состава модулей. Порядок изменения состава модулей приведен в документации	
13	Возможность отключения неиспользуемых модулей платформы	Отключить модуль upmeter и убедиться, что из кластера был удален namespace d8-upmeter	Модуль upmeter
14	Отказоустойчивая конфигурация всех компонентов платформы	При установке multi-master кластера убедиться, что служебные компоненты (prometheus, grafana, dex) находятся в двух репликах и распределены между узлами кластера	

15	Управление namespaces (добавление, удаление, редактирование)	Создать / удалить / добавить labels на произвольный namespace	
16	Возможность использования внешних модулей	Установить в Deckhouse Kubernetes Platform внешний оператор, например postgres-operator - убедиться в запуске подов оператора	
17	Аудит событий Kubernetes API	Настроен сбор Kubernetes audit log и в файле аудита записываются все действий в Kubernetes	Модуль control-plane-manager - Аудит
18	Фильтрации трафика внутри кластера (поддержка NetworkPolicy). Только для кластеров с CNI Cilium	Произвести настройку policyAuditMode и протестировать работу Network Policy	Модуль cni-cilium
19	Фильтрации трафика на уровне L7 внутри кластера (поддержка CiliumNetworkPolicy). Только для кластеров с CNI Cilium	Протестировать работу CiliumNetworkPolicy	Модуль cni-cilium
20	Отображения действия политик (NetworkPolicy) в веб-интерфейсе	Включить модуль cilium-hubble и проверить доступность web интерфейса, убедиться в наличии срабатываний networkPolicy	Модуль cilium-hubble
21	Возможность использования корпоративного TLS/SSL сертификата для	Средствами Deckhouse Platform заказан и успешно выпущен TLS/SSL сертификат с использованием корпоративного промежуточного	Модуль cert-manager

	компонентов платформы	сертификата.	
22	Использования временных статических пользователей в кластере	Создать статического пользователя и успешное его использование для входа в web-интерфейсы платформы	Модуль user-authn - Пример создания статического пользователя
23	Использование статических групп пользователей в кластере	Создать статическую группу пользователей и успешное ее использование для выдачи прав доступа в Kubernetes API	Модуль user-authn - Пример добавления статического пользователя в группу
24	Использование внешнего провайдера аутентификации (LDAP/AD/OIDC)	Включить модуль user-authn и настроить DexProvider. Убедиться, что есть возможность входа в web-интерфейсы Deckhouse Platform с использованием LDAP доступов	Модуль user-authn
25	Настройка ролевой модели доступа на основе групп, атрибутов пользователя	Выдать доступ пользователю по наличию в группе (LDAP / AD / OIDC) на основе ролевой модели доступа	Модуль user-authz - Пример ClusterAuthorizationRule
26	Ограничение доступа пользователей к определенным namespace	Выдать доступ пользователю/группе на заданный namespace - убедиться, что пользователь имеет права на работу с namespace	Как ограничить права пользователю конкретным namespace
27	Возможность расширения прав доступа	Расширить роль доступа User правами работы с секретами - убедиться, что у пользователя появились дополнительные права доступа	Модуль user-authz - Настройка прав высокоуровневых

			ролей
28	Использование сервисной учетной записи для выката прикладного ПО в платформу	Создать выделенную учетную запись с правами выката прикладного ПО в определенный namespace без доступа к другим namespace - произвести выкат приложения из под данной учетной записи	Создание ServiceAccount для сервера и предоставление ему доступа
29	Создание статического пользователя с помощью клиентского сертификата	Создать пользователя с помощью клиентского сертификата	Модуль user-authz - Создание пользователя с помощью клиентского сертификата
30	Использование политик безопасности Kubernetes (Pod Security Standards)	Применить политику Restricted и создать pod с privileged: true - такой под не должен быть создан	Модуль admission-policy-engine
31	Использование операционных политик для безопасной работы прикладного ПО	Применить OperationPolicy и создать pod с нарушением данной политики - такой под не должен быть создан	Модуль admission-policy-engine - Операционные политики
32	Использование политик безопасности для безопасной работы прикладного ПО	Применить SecurityPolicy и создать pod с нарушением данной политики - такой под не должен быть создан	Модуль admission-policy-engine - Политики безопасности

33	Возможность использовать квот в рамках namespaces	Создать ресурс ResourceQuota и поды, которые запрашивают ресурсов больше, чем выделено - такие поды не должны быть созданы	Resource Quota
34	Создание изолированного окружения по заготовленному шаблону	Создать проект из шаблона и проверить автоматическое создание ресурсов в заданном namespace	Модуль multitenancy-manager
35	Обнаружение угроз безопасности анализирую прикладное ПО и контейнеры	Запустить shell в контейнере и убедиться, что уведомление отработало корректно.	Модуль runtime-audit-engine - Добавление правила для отправки уведомлений о запуске shell-оболочки в контейнере
36	Организация mTLS между узлами прикладного ПО	Создать 2 сервиса с istio-proxy и убедиться, что взаимодействие между ними происходит с использованием mTLS	Модуль istio
37	Организация авторизации доступа между сервисами	Применить AuthorizationPolicy и ограничить взаимодействие между двумя сервисами. Убедиться, что неавторизованные запросы завершаются с ошибкой	Модуль istio, примеры использования
38	Сканирование образов прикладного ПО на наличие известных уязвимостей	Запустить контейнеры в платформе и убедиться в наличии результатов сканирования в CustomResource и в Grafana	Модуль operator-trivy