УДК 004.056.53 **Информационные технологии**

Липатова Софья Евгеньевна, *студент-магистр, Калужский филиал ФГБОУ ВО «Московский государственный технический университет имени Н.Э. Баумана» (национальный исследовательский университет)*

Белов Юрий Сергеевич, *к.ф.-м.н., доцент, Калужский филиал ФГБОУ ВО «Московский государственный технический университет имени Н.Э. Баумана» (национальный исследовательский университет)*

ПРАКТИКИ ОБЕСПЕЧЕНИЯ КИБЕРБЕЗОПАСНОСТИ В KUBERNETES

Аннотация: В настоящее время активно растет число микросервисных приложений, реализуемых с помощью технологии контейнеризации и платформ оркестровки, одной из которых является Kubernetes. При этом одним из наиболее важных вопросов остается обеспечение информационной безопасности системы, поэтому разработчикам требуется полный анализ и непосредственное понимание каждого из возможных случаев угрозы, зависимостей между элементами и вариантов защиты продукта. Kubernetes имеет встроенные системы защиты, однако в силу специфики приложений их может быть не достаточно, в связи с чем появляется потребность в применении дополнительных практик по обеспечению кибербезопасности.

Ключевые слова: информационная безопасность, приложение, Kubernetes, микросервисная архитектура.

Abstract: Currently, the number of microservice applications implemented using containerization technology and orchestration platforms, one of which is Kubernetes, is actively growing. At the same time, one of the most important issues remains ensuring the information security of the system, so developers need a complete analysis and direct understanding of each of the possible cases of threat, dependencies between elements and product protection options. Kubernetes has built-in security

systems, but due to the specifics of applications, they may not be enough, and therefore there is a need for additional practices to ensure cybersecurity.

Keywords: information security, application, Kubernetes, microservice architecture.

Введение. В настоящее время всё больше микросервисных приложений в полной мере реализуются с помощью контейнеризации – подхода, при котором приложение или служба, их зависимости и конфигурация упаковываются вместе в образ контейнера [1]. При этом возможно осуществить тестирование полученного единого продукта, развертывание которого происходит как экземпляр образа контейнера в операционной системе узла. При этом важно иметь возможность контроля и управления над процессом развертывания приложения. В связи с этим популярность набирают платформы оркестровки, одной из которых является Kubernetes.

Понятие Kubernetes. Kubernetes (K8s) — это портативная расширяемая платформа с открытым исходным кодом для управления контейнеризованными рабочими нагрузками и сервисами, которая облегчает как декларативную настройку, так и автоматизацию [2]. В настоящий момент популярность платформы набирает обороты как в России, так и по всему миру, в связи с чем её экосистема активно развивается. Кроме того, есть множество доступных сервисов и инструментов K8s.

Многие компании используют Kubernetes, потому что он сокращает повторяющиеся ручные процессы, связанные с развертыванием контейнеров и управлением ими. K8s считается одним из самых популярных инструментов оркестрации контейнеров с открытым исходным кодом и используется в таких организациях, как Adidas, Nokia, Spotify и т.д. Так, например, использование Kubernetes в компании Adidas сократило время загрузки веб-сайта электронной коммерции вдвое, а частота релизов увеличилась с одного раза в 4–6 недель до 3–4 раз в день [3].

Однако не смотря на преимущества системы, в силу её специфики много вопросов и беспокойств вызывает её кибербезопасность. Согласно исследованиям Фонд Cloud Native Computing Foundation, 40% участников опроса обеспокоены безопасностью Kubernetes [4]. И на это есть причины: например, в 2018 году злоумышленники получили доступ к ресурсам Tesla Amazon Web Services (AWS) с помощью небезопасной консоли Kubernetes [5]. Важно отметить, что группа разработки Kubernetes активно борется с проблемами безопасности, стараясь максимально выявлять и исправлять возможные «бреши». Уже сейчас реализовано множество способов обеспечения кибербезопасности данной платформы.

Практики обеспечения безопасности в Kubernetes. Рассмотрим основные практики и способы обеспечения кибербезопасности в Kubernetes:

1) Аутентификация и авторизация.

Здесь важно выделить возможности применения правил аутентификации и авторизации для предотвращения доступа злоумышленников и выполнения несанкционированных действий внутри кластера Kubernetes. Аутентификация в K8s относится к аутентификации запросов API через плагины аутентификации, в то время как авторизация относится к оценке каждого аутентифицированного запроса API на соответствие всем политикам для разрешения или отклонения запроса [2].

Действия для реализации данной практики:

— Необходимо отключить анонимный доступ к серверу Kubernetes, разрешенный системой по умолчанию [2].

— Необходимо отключить режимы авторизации по умолчанию.

— Контроллеры доступа, представляющие собой инструменты, перехватывающие запросы к API K8s после проверки подлинности и авторизации запроса, но до того, как том станет постоянным, должны быть включены.

— Необходимо настроить управление использованием олицетворения. Функция олицетворения имеет свои преимущества, однако в случае

невозможности определить ограничения на то, кто может олицетворять и что может делать олицетворяемый пользователь, функция олицетворения может нанести ущерб безопасности Kubernetes.

— Конфигурации по умолчанию должны быть изменены. Использование конфигурации по умолчанию для аутентификации и авторизации может позволить любому анонимному неаутентифицированному пользователю выполнять вредоносные действия.

Для аутентификации и авторизации можно использовать стандартный протокол аутентификации OpenID, а также есть возможность прмиенения веб-перехватчиков, управления доступом на основе ролей (RBAC) и управления доступом на основе атрибутов (ABAC) [2].

2) Внедрение специфичных для Kubernetes политик безопасности.

Рассмотрим различные политики в K8s и способы обеспечения безопасности в них:

— Сетевые политики. По умолчанию все модули Kubernetes могут взаимодействовать с другими модулями, именно поэтому можно использовать политику ограничения трафика между подами, установить надлежащие брандмауэры для блокировки всех нежелательных сетевых коммуникаций и настроить ограниченный доступ к базе данных для модулей. В противном случае любой злоумышленник может атаковать сервер API с любого IP-адреса.

— Политики, специфичные для пода. Без определения безопасного контекста для модуля контейнер может работать с привилегиями root и разрешением на запись в корневую файловую систему, что может сделать кластер Kubernetes уязвимым. В связи с этим важно, чтобы контейнеры внутри пода запускались не от имени root-пользователя, с правами только на чтение и включенными модулями безопасности Linux. Также возможна установка минимальных версий операционных систем для уменьшения поверхности атаки.

— Общие политики. TCP-порты для kubelet, API-сервера и т.д., а также сетевых плагинов не должны оставаться открытыми и обязаны требовать аутентификации для обеспечения видимости. По умолчанию каждый

пользователь в системе должен иметь наименьшие привилегии. Публичный доступ SSH к узлам кластера Kubernetes должен быть ограничен. Пользователям K8s рекомендуется создать для ведения журналов политику аудита, настроенную для каждого кластера Kubernetes на уровне сервера API.

3) Сканирование уязвимостей.

Компоненты Kubernetes, такие как контейнеры, могут содержать уязвимости и вредоносные программы. Если в кластере Kubernetes присутствуют уязвимости, то вся система оркестрации контейнеров и подготовленные приложения становятся уязвимыми для атак. Поэтому рекомендуется сканировать контейнеры на наличие уязвимостей с помощью таких инструментов, как «Dockscan» и «CoreOS Clair». При этом если не проверять образы и конфигурации развертывания в компонентах компакт-диска, это может сделать кластер Kubernetes уязвимым для злоумышленников. Поэтому важно извлекать образы из доверенного частного реестра и проверять код и образы на уязвимость.

4) Ведение журнала (логирование).

Без включения ведения журналов и мониторинга пользователи могут столкнуться с трудностями при устранении неожиданных последствий, таких как атаки со стороны злоумышленников и сбои в работе. Для реализации практики ведения журналов возможны следующие решения:

— Журналы должны отслеживаться через регулярные промежутки времени.

— Необходимо настроить оповещения для любых резких изменений показателей журнала по сравнению с предыдущими записями журнала.

5) Разделение пространств имен.

Пространство имен в Kubernetes — это логически изолированный виртуальный кластер внутри одного физического кластера [2]. Создание отдельных пространств имен позволяет изолировать ресурсы между пространствами имен. Если для ресурса не создается отдельное пространство имен, ресурс получает пространство имен «по умолчанию». Поэтому важно,

чтобы у каждой команды в компании было отдельное пространство имен для лучшей управляемости и запуска сред разработки и производства. Если есть только пространство имен «по умолчанию» и нет отдельного пространства имен для разных команд, то любой злоумышленник может выполнить атаку на пространство имен «по умолчанию», сделав весь ресурс уязвимым для этой атаки. На практике для данного решения используют флаг –namespace в команде kubectl для разделения пространств имен.

6) Шифрование и ограничение доступа к etcd (внутренней базе данных, используемой Kubernetes).

Разработчики рекомендуют, чтобы «etcd» был доступен только с серверов API и был изолирован за брандмауэром, чтобы посторонние не могли получить доступ через API [2]. По умолчанию Kubernetes хранит секретные данные в виде открытого текста в файле etcd, завладев которым можно получить конфиденциальную информацию, такую как имена пользователей базы данных, пароли и запросы. Хотя Kubernetes шифрует «etcd», ключ для шифрования хранится в виде открытого текста в файле конфигурации на главном узле. По этой причине рекомендуется использовать инструменты управления секретами для дополнительной безопасности [2], такие как Vault для шифрования.

7) Непрерывное обновление.

Пользователям Kubernetes рекомендуется применять обновления, а также проводить постоянные обновления для развернутых приложений в модулях Kubernetes. Без постоянных обновлений в установке Kubernetes могут существовать уязвимости, которые могут дать злоумышленникам возможность проводить атаки. Для непрерывных обновлений также рекомендуется использовать последовательное обновление, т. е. устанавливать исправления Kubernetes без нарушения доступности развернутых приложений. K8s предоставляет такие инструменты, как «kubectl», для выполнения последовательных обновлений [2].

8) Ограничение квоты ЦП и памяти.

По умолчанию все ресурсы в Kubernetes начинаются с неограниченных запросов/ограничений памяти и неограниченного доступа к ЦП. Если злоумышленник запускает атаку типа «отказ в обслуживании» (DOS) в модуле в кластере Kubernetes, то из-за большого объема запросов kube-scheduler создаст новый модуль, и экземпляр контейнера запустится внутри нового модуля узла. Этот процесс будет продолжаться до тех пор, пока он не израсходует все доступные ресурсы ЦП и памяти, в результате чего все приложения будут «голодать». Следовательно, неспособность определить лимиты запросов ЦП и памяти для пода или пространства имен может привести к потреблению всех доступных ресурсов в кластере Kubernetes. Поэтому можно настроить количество ресурсов, определив максимальное количество экземпляров для контейнера, количество ресурсов ЦП, потребляемых приложением, и максимальный объем памяти для модуля или пространства имен, что позволяет «смягчить» вредоносные атаки.

9) Включение поддержки SSL/TLS.

Включение TLS между сервером API kubernetes, etcd, kubelet и kubectl обеспечивает безопасную связь между компонентами кластера. Поэтому хорошим решением является включение сертификатов TLS и SSL для компонентов Kubernetes.

10) Отдельная конфиденциальная рабочая нагрузка.

Здесь рассматривается практика запуска конфиденциальных приложений на выделенном наборе компьютеров для ограничения потенциального воздействия нарушения безопасности. Рекомендуется использовать предоставляемые Kubernetes утилиты, такие как «taints and tolerations», которые могут контролировать, где может быть развернут модуль.

11) Безопасный доступ к метаданным.

API-интерфейсы метаданных Kubernetes предоставляют шлюз для предоставления учетных данных администратора «kubelet». Google рекомендует активировать такие функции, как Workload Identity для Google Kubernetes Engine

(GKE), чтобы предотвратить утечку конфиденциальной информации через службу метаданных [2].

Таким образом, представленные выше способы обеспечения безопасности внутри Kubernetes являются рекомендуемыми при работе с кластером K8s, при это выбор используемых практик зависит от специфики кластера.

Заключение. В настоящее время Kubernetes становится всё популярнее по всему миру, а его экосистема стремительно растет. Ввиду специфики самой платформы, важно уделять внимание информационной безопасности. Выделено достаточно много способов и рекомендаций по обеспечению кибербезопасности данных и приложений в K8s. При этом каждый способ может быть использован как отдельно, так и в совокупности с остальными в зависимости от потребностей разработки. При использовании K8s важно уметь увидеть возможные уязвимости и суметь закрыть их раньше, чем возникнет угроза со стороны злоумышленников. При этом комьюнити Kubernetes растет, и в случае возникновения проблем или сомнений всегда можно обратиться к экспертам с целью получения помощи или рекомендации. Чем больше будет понимания со стороны разработки в сфере требуемых настроек, возможностей, инструментов, тем проще будет увидеть возможные слабые стороны системы и защитить их.

Библиографический список:

1. Маркелов А. А. Введение в технологию контейнеров и Kubernetes. — Москва: ДМК Пресс, 2019. — 194 с.

2. Документация по Kubernetes [Электронный ресурс] URL: https://kubernetes.io/ru/docs/home/ (дата обращения 25.12.2021).

3. With Kubernetes, the U.S. Department of Defense Is Enabling DevSecOps on F-16s and Battleships [Электронный ресурс] URL: https://www.cncf.io/case-study/dod/ (дата обращения 25.12.2021).

4. CNCF. CNCF SURVEY 2019 [Электронный ресурс] URL: https://www.cncf.io/wp-content/uploads/2020/08/CNCF_Survey_Report.pdf (дата обращения 25.12.2021).

5. Dan Goodin. Tesla cloud resources are hacked to run cryptocurrency-mining malware [Электронный ресурс] URL: https://arstechnica.com/information-technology/2018/02/tesla-cloud-resources-are-hacked-to-run-cryptocurrency-mining-malware/ (дата обращения 25.12.2021).