УДК 004.457

# ОСНОВНЫЕ ПОКАЗАТЕЛИ ДОСТУПНОСТИ В KUBERNETES И СПОСОБЫ ЕЕ ОБЕСПЕЧЕНИЯ

**Липатова С.Е., Белов Ю.С.**

*Московский государственный технический университет имени Н.Э. Баумана, Калужский филиал, Калуга, e-mail: fn1-kf@mail.ru*

Kubernetes – один из популярнейших инструментов оркестровки контейнеров, обеспечивающий возможность гибкого управления системой. Постепенно внедряясь в компании разработки по всему миру, одним из ключевых аспектов данной технологии остается доступность. Kubernetes позволяет обеспечить более высокую степень доступности приложений по сравнению с аналогами в различных сценариях сбоев. В данном исследовании будут рассмотрены как особенности работы Kubernetes и его внутреннее устройство для полного понимания процессов, так и метрики доступности, позволяющие проанализировать состояние системы и её поведение в различных ситуациях. Помимо этого, будут рассмотрены уровни проверки работоспособности Kubernetes, исходя из которых можно выделить основные сценарии сбоев и способы их моделирования для проведения анализа эффективности системы, оптимальность настроек для обеспечения доступности, а также самостоятельного исследования степени защищенности и стабильности системы, её поведения во внештатных ситуациях. Всё это играет важную роль как для небольших проектов, так и для крупных и сложных систем, уровень доступности является одним из ключевых параметров стабильности сервисов, что напрямую влияет на техническую среду, экономические показатели, а также уровень доверия и одобрения пользователей. Чем выше скорость восстановления системы после сбоев и отказоустойчивость сервисов, тем выше показатели работы приложения в разрезе различных сфер и компонентов.

**Ключевые слова: Kubernetes, доступность, метрики доступности, сценарии сбоев**

# THE MAIN INDICATORS OF AVAILABILITY IN KUBERNETES AND WAYS TO ENSURE IT

**Lipatova S.E., Belov Yu.S.**

*Bauman Moscow State Technical University, Kaluga branch, Kaluga, e-mail: sonya_lipatova@list.ru*

Kubernetes is one of the most popular container orchestration tools, providing flexible system management. Gradually being introduced into development companies around the world, one of the key aspects of this technology remains accessibility. Kubernetes allows you to provide a higher degree of application availability compared to analogues in various failure scenarios. This study will examine both the features of Kubernetes and its internal structure for a complete understanding of processes, and availability metrics that allow analyzing the state of the system and its behavior in various situations. In addition, the levels of Kubernetes health check will be considered, based on which it is possible to identify the main failure scenarios and ways to model them for analyzing the effectiveness of the system, the optimality of settings to ensure availability, as well as an independent study of the degree of security and stability of the system, its behavior in emergency situations. All this plays an important role both for small projects and for large and complex systems, the level of availability is one of the key parameters of service stability, which directly affects the technical environment, economic indicators, as well as the level of trust and approval of users. The higher the speed of system recovery after failures and the fault tolerance of services, the higher the performance of the application in the context of various spheres and components.

**Keywords: Kubernetes, availability, availability metrics, failure scenarios**

В связи с ростом популярности технологии Kubernetes, а также микросервисной архитектуры, имеющей множество преимуществ в сравнении с монолитной, важно уделить внимание одному из ключевых факторов использования этого стека: доступности микросервисов внутри кластеров, поскольку от этого напрямую зависит как стабильность сервисов, реакция пользователей, имеющая ведущую роль в работе приложений и систем, а также прибыль компаний-разработчиков. Однако перед тем, как рассмотреть критерии доступности и её составляющие, необходимо рассмотреть саму технологию Kubernetes и внутреннее устройство кластера для понимания особенностей работы данной технологии, а также определить, что именно относится к понятию «доступность».

Цель исследования – выделить основные показатели доступности в Kubernetes на уровне метрик, уровней проверки работоспособности и сценариев сбоев, а также выявить способы её обеспечения.

*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 [1].

В настоящий момент всё больше компаний по всему миру переходят к использованию данной технологии ввиду того, что это

позволяет сократить повторяющиеся ручные процессы, связанные с развертыванием контейнеров и управлением ими. С ростом популярности технологии растёт и её экосистема, что позволяет быстро найти ответы на интересующие вопросы по настройке конфигурации системы и т.д.

Являясь одним из самых популярных инструментов оркестровки контейнеров с открытым исходным кодом, Kubernetes стремительно проникает в стек технологий крупнейших компаний мира. Однако это не просто система оркестровки. Фактически он устраняет необходимость в оркестровке, поскольку включает в себя набор независимых, составляемых процессов управления, которые непрерывно приводят текущее состояние к заданному желаемому состоянию [2].

По своей сути Kubernetes представляет собой один из стратегических компонентов всего DevOps-процесса [3], именно поэтому большое внимание уделяется как кибербезопасности, так и доступности, поскольку именно эти два параметра имеют решающее значение как для функционирования сервисов в целом, так и для экономических и бизнес-показателей.

Иными словами, Kubernetes – это система гибкого управления инфраструктурой контейнеризации с возможностью балансировки нагрузки, обеспечивающая скорость, гибкость технологии и экономическую эффективность [4].

*Устройство кластера Kubernetes.* Рассмотрим устройство кластера Kubernetes для полного понимания внутренних процессов и компонентов.

Основные компоненты кластера Kubernetes представлены на рис. 1.

Компоненты панели управления отвечают за основные операции кластера (например, планирование), а также обрабатывают события кластера (например, запускают новый под, когда поле replicas развертывания не соответствует требуемому количеству реплик) [1].

Рассмотрим подробнее компоненты управления K8s:

1) kube-apiserver – сервер API – компонент Kubernetes панели управления, который представляет API Kubernetes. API-сервер – это клиентская часть панели управления Kubernetes. Он предназначен для горизонтального масштабирования. Есть возможность запуска нескольких экземпляров kube-apiserver и балансировки трафика между ними;

2) etcd – распределённое и высоконадёжное хранилище данных в формате «ключ-значение», которое используется как основное хранилище всех данных кластера в Kubernetes;

3) kube-scheduler – компонент плоскости управления, который отслеживает созданные поды без привязанного узла и выбирает узел, на котором они должны работать;

4) kube-controller-manager – компонент Control Plane запускает процессы контроллера;

5) cloud-controller-manager – запускает контроллеры, которые взаимодействуют с основными облачными провайдерами [1].

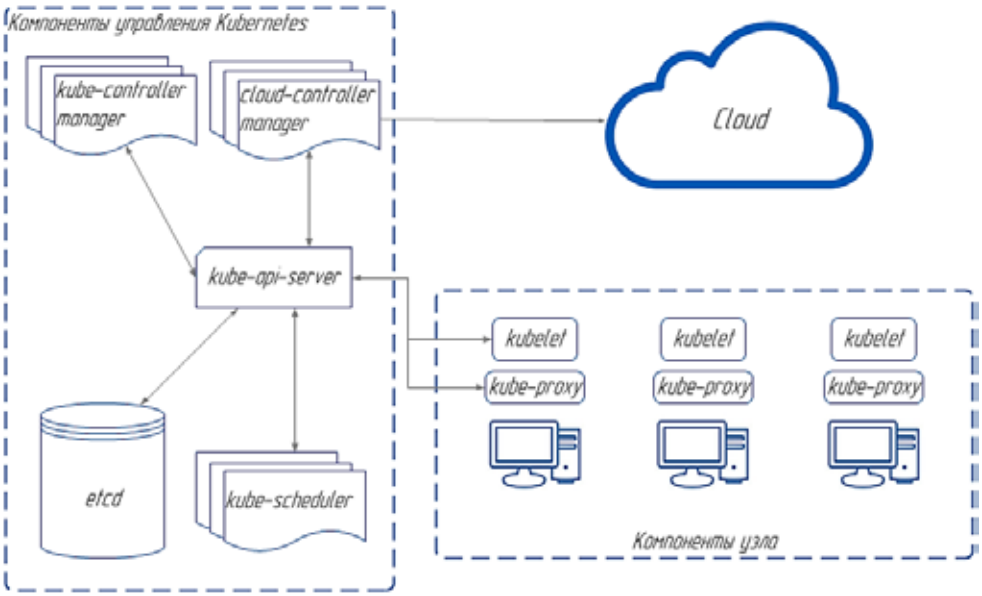


*Рис. 1. Компоненты кластера Kubernetes*

Рассмотрим компоненты узла, поддерживающие работу подов и среды выполнения Kubernetes:

1) kubelet – утилита на каждом кластере, следящая за тем, чтобы контейнеры были запущены в поде;

2) kube-proxy – сетевой прокси, работающий на каждом узле в кластере и реализующий часть концепции сервис. Он конфигурирует правила сети на узлах;

3) среда выполнения контейнеров [1].

К дополнениям, расширяющим возможности K8s, относят DNS, Dashboard (веб-интерфейс), мониторинг ресурсов и логирование кластера.

*Доступность.* Доступность услуги – это нефункциональное требование, определяемое как процент времени предоставления услуги [5]. Высокая доступность достигается, когда система доступна не менее 99,999% времени, т.е. общее допустимое время простоя высокодоступных систем – около 5 минут в год [2].

Важной особенностью Kubernetes является перезапуск, замена или переназначение неисправных контейнеров в случае сбоя их хостов. Объявление контейнеров снова работоспособными происходит лишь после полного цикла их подготовки. Но эти меры могут оказаться недостаточными для поставщиков услуг операторского класса, и доступность в качестве важного нефункционального требования по-прежнему вызывает у них беспокойство [6]. Для этого требуется дополнительный анализ каждой системы, поскольку в зависимости от приложений, их назначений и сервисов настройки K8s будут различаться, а, следовательно, показатели доступности могут разниться.

*Метрики доступности.* Рассмотрим метрики, которые используются при анализе доступности Kubernetes и их зависимость между собой. Схематично они представлены на рис. 2.

Рассмотрим подробнее каждую из метрик:

1) время реакции Kubernetes на сбой – период между сбоем системы и первой реакцией Kubernetes, отражающей его обнаружение;

2) время исправления модуля – период между первой реакцией Kubernetes на сбой и моментом восстановления модуля;

3) время восстановления службы – период между первой реакцией Kubernetes на сбой и моментом возобновления доступности службы;

4) время простоя – общая продолжительность времени реакции и времени восстановления доступности, в течение которого услуга была недоступна [7].

Анализ данных метрик на этапах настройки, изменения или отладки конфигурации способен помочь добиться максимальных условий для обеспечения высокого уровня доступности приложений, развернутых в K8s.

*Уровни проверки работоспособности Kubernetes.* Рассматривая возможные сценарии сбоев, важно отметить, что Kubernetes предлагает три уровня проверки работоспособности и действий по исправлению для управления доступностью развернутых приложений:

1) прикладной уровень. На данном уровне Kubernetes гарантирует исправность программных компонентов, выполняющихся внутри контейнера, с помощью проверки работоспособности процесса или предопределенных тестов. В случае, если Kubelet обнаружит сбой, он отреагирует в соответствии с определенной политикой перезапуска;

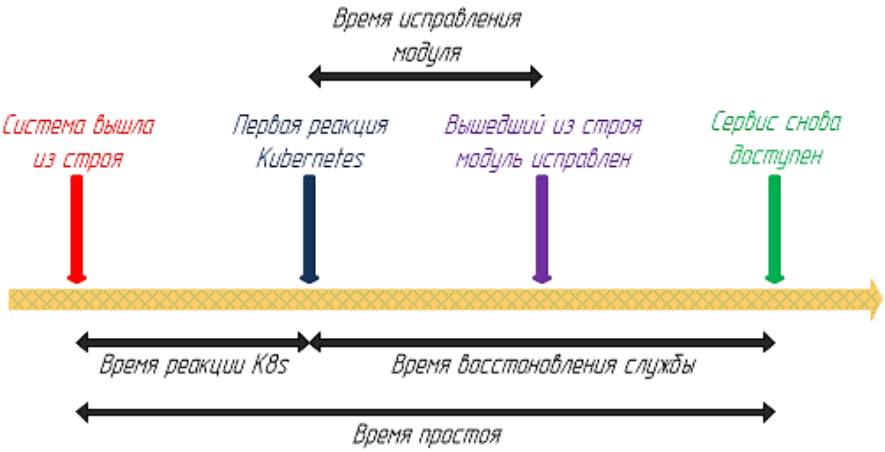


*Рис. 2. Метрики доступности K8s при сбое*

2) уровень процесса пода. Здесь Kubernetes отслеживает сбои внутри пода, являющегося средой, предоставляемой для запуска контейнеров приложений, путем предоставления общего хранилища и сети для них.

3) уровень узла (ноды). В данном случае Kubernetes отслеживает состояние узлов кластера через свой компонент node controller. Если узел, на котором размещен модуль, выходит из строя, модуль переносится на другой исправный узел.

Данные уровни показывают уязвимые места системы, что важно учитывать при настройке K8s.

*Сценарии сбоев.* Исходя из уровней проверки работоспособности, можно выделить следующие сценарии сбоев и возможности их воспроизведения для настройки конфигурации:

1) отключение службы из-за сбоя контейнера приложения – для имитации данного сбоя достаточно завершить процесс работы контейнера приложения из операционной системы.

2) сбой в обслуживании из-за сбоя процесса пода – моделирование сбоя возможно с помощью завершения процесса пода из операционной системы (ОС), поскольку при развертывании пода вместе с контейнерами приложений, указанными в его шаблоне, создается один дополнительный контейнер, который является процессом пода. Поскольку модуль сам по себе является процессом в ОС, есть вероятность, что он выйдет из строя [7];

3) сбой в обслуживании из-за нарушения работы узла – узел, на котором размещен модуль, выходит из строя. Можно смоделировать командой перезагрузки Linux или путем выключения узла.

Все эти сбои имеют место в работе любой системы, именно поэтому важно знать их роль в доступности приложения на основе K8s и рассмотреть встроенные возможности по их устранению.

*Способы обеспечения доступности с помощью Kubernetes.* Рассмотрим основные блоки возможных действий Kubernetes при сбое, которые способны помочь системе вернуться к работоспособности:

1) действия по восстановлению с использованием конфигурации Kubernetes по умолчанию. Сама технология Kubernetes повышает доступность системы путем настройки конфигурации при разворачивании технологии;

2) добавление избыточных экземпляров. Зачастую некоторые разработчики приходят к решению добавить избыточные экземпляры для повышения доступности, однако важно проанализировать данное решение на этапе разворачивания системы, поскольку это может негативно сказаться на ресурсах и спровоцировать новые риски;

3) восстановление путем использования наиболее гибкой конфигурации K8s.

При настройке Kubernetes есть возможность для указания параметров, способных повлиять на восстановление, например ускорить его. Здесь важно верно учесть ресурсы, внутренние тайминги системы, специфику приложений и микросервисов. Таким образом, система будет использовать наиболее гибкую конфигурацию, что положительно скажется на доступности сервисов.

Помимо этого, важно учитывать средние показатели внутренних таймингов Kubernetes:

1) сигнал завершения, отправляемый в контейнер приложения в сценарии сбоя процесса пода, занимает более 30 с;

2) при сценарии сбоя узла частота отправки статуса узла Kubelet ведущему устройству составляет 10 с, а количество разрешенных пропущенных обновлений статуса перед пометкой узла как неработоспособного равно четырем, что составляет время реакции от 30 до 40 с [8];

3) при выходе модуля из строя и перед созданием нового Kubernetes ожидает около 260 с [8].

Данные тайминги необходимо учитывать, как и остальные аспекты настройки, сценарии, риски и специфики, для настройки K8s, способной обеспечить максимальные показатели доступности.

В настоящий момент Kubernetes является одной из ключевых технологий оркестровки приложений. Одним из факторов роста количества его внедрений является возможность повышения доступности сервисов, контроль их работы. Внутри K8s можно выделить 3 уровня работоспособности, напрямую коррелирующие с возможными сценариями сбоев: прикладной уровень, процесса пода и процесса ноды. При этом разработчик может влиять на процесс восстановления работоспособности сервисов на данных уровнях путем настройки конфигурации по умолчанию, добавления избыточных экземпляров или использования более гибкой конфигурации K8s, но важно отметить, что каждое из решений имеет свои риски, зависящие от приложения, которые должны быть проанализированы перед настройкой системы и в процессе отладки конфигурации.

**Список литературы**

1. Документация по Kubernetes. [Электронный ресурс]. URL: https://kubernetes.io/ru/docs/home/ (дата обращения: 25.05.2022).

2. Sebrechts M., Borny S., Wauters T., Volckaert B. Service Relationship Orchestration: Lessons Learned From Running Large Scale Smart City Platforms on Kubernetes // IEEE Access. 2021. Vol. 9. P. 133387–133401.

3. Липатова С.Е., Белов Ю.С. Практики обеспечения кибербезопасности в Kubernetes // E-Scio. 2022. № 1. С. 490–498.

4. Han J., Hong Y., Kim J. Refining Microservices Placement Employing Workload Profiling Over Multiple Kubernetes Clusters. IEEE Access. 2020. Vol. 8. P. 192543–192556.

5. Sebastio S., Ghosh R., Mukherjee T. An Availability Analysis Approach for Deployment Configurations of Containers. IEEE Transactions on Services Computing. 2021. Vol. 14. No. 1. P. 16–29.

6. Липатова С.Е., Белов Ю.С. Принцип работы контроллера statefulset Kubernetes для управления доступностью приложений с отслеживанием состояния на основе микросервисов // Высокие технологии и инновации в науке: сборник избранных статей Международной научной конференции, 2022. С. 112–115.

7. Qi S., Kulkarni S.G., Ramakrishnan K.K. Assessing Container Network Interface Plugins: Functionality, Performance, and Scalability. IEEE Transactions on Network and Service Management. 2021. Vol. 18. No. 1. P. 656–671.

8. Taherizadeh S., Stankovski V., Cho J. Dynamic Multi-level Auto-scaling Rules for Containerized Applications. The Computer Journal. 2019. Vol. 62. No. 2. P. 174–197.