ТЕХНИЧЕСКОЕ ОПИСАНИЕ

10 УРОВНЕЙ КОНТЕЙНЕРНОЙ БЕЗОПАСНОСТИ

ВВЕДЕНИЕ

Контейнеры пользуются высокой популярностью, поскольку позволяют легко упаковать приложение вместе со всеми зависимостями в один образ, который разработчики могут тут же передать специалистам по тестированию и развертыванию без каких-либо изменений. Обеспечивая переносимость между средами и поддерживая развертывание на физических серверах, виртуальных машинах, в частных и общедоступных облаках, контейнеры значительно упрощают разработку и управление приложениями, принося ощутимую выгоду.

Корпоративный сектор, традиционно имеющий высокие требования к безопасности, в первую очередь интересует, насколько безопасны контейнеры, и можно ли доверить им ключевые сервисы и приложения. В этом документе мы рассмотрим 10 ключевых аспектов защиты контейнеров на различных уровнях контейнерной архитектуры и этапах жизненного цикла приложения.

redhat.com

БЕЗОПАСНОСТЬ КОНТЕЙНЕРОВ: АРХИТЕКТУРНЫЕ УРОВНИ И ЖИЗНЕННЫЙ ЦИКЛ

В плане безопасности контейнеры имеют много общего системными процессами. Прежде чем приступать к развертыванию и запуску контейнеров, необходимо продумать безопасность на всех уровнях архитектуры и этапах жизненного цикла контейнеров и приложений. Список ключевых аспектов безопасности контейнеров содержит 10 пунктов:

1. Мультитенантность контейнерного хоста;
2. Содержимое контейнера;
3. Реестры контейнеров;
4. Сборка контейнеров;
5. Развертывание контейнеров;
6. Оркестрация контейнеров;
7. Сетевая изоляция;
8. Хранилища;
9. Управление API-интерфейсами;
10. Федеративные кластеры.

1. Операционная система контейнерного хоста и мультитенантность

Для разработчика контейнер – это простой способ упаковать приложение вместе с зависимостями в готовый к использованию и удобный для распространения модуль. Для оператора ИТ-систем контейнер – это возможность повысить производительность серверов и плотность размещения приложений за счет мультитенантного развертывания, иначе говоря, разместить на хосте несколько контейнеров и запускать и останавливать их по мере надобности. Почти как виртуальные машины, только без гостевой ОС и гипервизора, поскольку предметом виртуализации здесь является не «железо», а операционная система.

Для реализации преимуществ контейнерной упаковки и развертывания приложений требуется соответствующая среда выполнения – операционная система, надежно изолирующая контейнеры друг от друга и от своего ядра.

Контейнер представляет собой группу изолированных процессов операционной системы Linux, ограниченных в использовании системных ресурсов. Это позволяет организовать для каждого контейнерного приложения свою собственную «песочницу» на основе общего для всех ядра ОС хоста. С точки зрения безопасности контейнеры идентичны процессам. Для них так же важно и действенно ограничивать полномочия, желательно, до строго необходимого минимума, и запускать от непривилегированного пользователя (не с правами root). Кроме того, пригодятся и общие средства безопасности Linux, в частности, Red Hat® Enterprise Linux предлагает пять таких средств:

- **Пространства имен Linux** – фундамент всей системы изоляции контейнеров. Абстрагирование на этом уровне создает у процессов иллюзию монопольного доступа к операционной системе и ресурсам хоста;
- **SELinux** – дополнительный механизм изоляции контейнеров друг от друга и ОС хоста за счет мандатного управления доступом для каждого пользователя, приложения, процесса или файла. SELinux надежно пресекает любые, намеренные или случайные, попытки выйти за пределы абстракции пространства имен;

- **Группы управления Cgroups** – изоляция, квоты и биллинг для наборов **процессов** при использовании ресурсов хоста (процессор, память, дисковый ввод-вывод, сеть). Помогают предотвратить захват системных ресурсов одним контейнером в ущерб остальными;
- **Наборы ограничения полномочий (Linux capabilities)** – ограничения прав root внутри контейнера путем отзыва отдельных полномочий (capabilities), например, возможности отправлять «необработанные» (raw) IP-пакеты или выполнять привязку к портам ниже 1024-го. Для большинства контейнеризованных приложений можно безболезненно отключить целый ряд таких полномочий;
- И наконец, **seccomp** (secure computing mode) – механизм ограничения доступных контейнеру системных вызовов.

Дополнительно усилить безопасность контейнерных приложений и ИТ-инфраструктуры можно с помощью специализированных дистрибутивов Linux, таких как Red Hat Enterprise Linux Atomic Host, который оптимизирован для работы с контейнерами и содержит строго необходимый минимум пакетов, значительно сужая фронт потенциальных атак.

Надежность средств безопасности Red Hat Enterprise Linux и Atomic Host подтверждается сертификатом Common Criteria с поддержкой Linux Container Framework, который недавно был получен для ОС Red Hat Enterprise Linux 7.1.

Традиционная виртуализация тоже реализует мультитенантность, но совершенно иначе. Гипервизор на хосте виртуализации инициализирует виртуальные машины (ВМ), каждая из них загружает собственную ОС, которая уже и запускает приложение вместе с вспомогательными компонентами. За изоляцию ВМ друг от друга и от хоста отвечает гипервизор, доступ к которому имеет ограниченный круг лиц и процессов. Это тоже сокращает фронт атак, но не гарантирует 100 % безопасность. Например, из-за уязвимости в гипервизоре виртуальная машина может получить доступ к другим ВМ или к операционной системе хоста. Кроме того, если уязвимость обнаруживается в гостевой ОС, патчить приходится все ВМ, на которых она установлена.

Контейнеры тоже можно запускать на ВМ, и зачастую так и делают. Например, при переносе традиционных приложений в облако их упаковывают в контейнер и размещают этот контейнер на облачной ВМ. Однако по сравнению с традиционной виртуализацией мультитенантность на основе контейнеров требует меньше ресурсов и предлагает больше гибкости и масштабируемости при развертывании приложений на хосте, особенно в случае распределенных приложений.

2. Содержимое контейнера: пользуйтесь только надежными источниками

С точки зрения безопасности важно и то, что находится внутри контейнера. ИТ-инфраструктуры и программные решения уже давно строятся на основе готовых компонентов, включая программные пакеты с открытым кодом, такие как Linux, Apache Web Server, Red Hat JBoss® Enterprise Application Platform, PostgreSQL, Node.js и другие. Многие из этих пакетов сегодня доступны в виде контейнеров. Но, как и с любым сторонним ПО, важно знать, откуда взялись эти контейнеры, кто их собрал, и не содержат ли они вредоносного кода. Поэтому прежде чем использовать контейнер, спросите себя:

- Не опасно ли содержимое контейнера для моей ИТ-инфраструктуры?
- Не содержит ли оно известные уязвимости на уровне приложений?
- Используются ли там свежие версии ОС и среды выполнения?
- Часто ли обновляется этот контейнер, и как я узнаю об очередном обновлении?

Имея за плечами многолетний опыт производства и поставки надежного и проверенного софта в традиционной форме, Red Hat сегодня предлагает его в виде контейнеров. В каталоге Red Hat Container Catalog можно найти ОС Red Hat Enterprise Linux 7 и Red Hat Enterprise Linux 6, а также целый ряд сертифицированных runtime-компонентов для различных языков программирования, связующее ПО, СУБД и множество других продуктов и решений, представленных в виде контейнерных образов. Сертифицированные контейнеры Red Hat работают везде, где работает Red Hat Enterprise Linux – на «голом железе», на ВМ, в облаке, – и обеспечиваются поддержкой Red Hat и компаний-партнеров.

Red Hat также разработала и регулярно обновляет рейтинг безопасности контейнеров в каталоге Red Hat Container Catalog, получивший название Container Health Index. Этот индекс ранжирует опасность контейнера по наличию непропатченных уязвимостей, причем с учетом как критичности этих уязвимостей для компонентов контейнера, так и числа дней, прошедших с момента выпуска соответствующих исправлений безопасности.

При выпуске исправлений безопасности, наподобие патчей для glibc, Drown или Dirty Cow, мы оперативно обновляем свои образы в каталоге Red Hat Container Catalog. Кроме того, раздел Security Advisories на нашем портале Customer Portal оперативно информирует заказчиков о появлении уязвимостей в ранее выпущенных сертифицированных контейнерах и подсказывает, где скачать обновленные версии.

Разумеется, Red Hat Container Catalog не сможет закрыть все потребности в контейнерах, и вы будете брать их из других источников. В таких случаях мы рекомендуем применять сканеры безопасности контейнеров с постоянно обновляемой БД уязвимостей, чтобы быть в курсе рисков и действовать осознанно. Кроме того, с течением времени безопасность контейнера деградирует, поскольку после его выпуска, как правило, всегда обнаруживаются новые уязвимости. Поэтому вам следует регулярно проводить переоценку безопасности используемых контейнеров, как это делаем мы для контейнеров в Red Hat Container Catalog.

API-интерфейсы Red Hat Enterprise Linux позволяют подключать различные сканеры безопасности, такие как OpenSCAP, Black Duck Hub, JFrog Xray и Twistlock. Red Hat CloudForms также поддерживает проверку безопасности контейнеров с помощью OpenSCAP, а Red Hat OpenShift дает возможность задействовать сканеры безопасности в процессах непрерывной интеграции и доставки (CI/CD), ниже мы рассмотрим это подробнее.

3. Реестры контейнеров: защищенный доступ к контейнерным образам

При сборке собственных контейнеров вы будете использовать в качестве базового слоя общедоступные контейнерные образы. Управлять распространением и доступом к таким образам нужно так же, как и при работе с любыми другим бинарниками. Хранение контейнерных образов поддерживается в целом ряде частных репозиториев, но мы советуем выбирать такие, что позволяют автоматизировать политики использования образов.

В Red Hat OpenShift есть встроенный частный реестр с ролевой моделью доступа, позволяющий эффективно управлять контейнерными образами и контролировать, кто может помещать и извлекать их из реестра. OpenShift также поддерживает интеграцию с внешними частными реестрами, включая JFrog Artifactory и Docker Trusted Registry, на тот случай, если вы уже пользуетесь ими.

Список известных уязвимостей постоянно растет, поэтому содержимое как развернутых, так и недавно загруженных извне образов контейнеров следует регулярно проверять на предмет безопасности. И реестр должен помогать вам в этом, основываясь на метаданных образа, включая сведения о уязвимостях. Например, в Red Hat CloudForms есть функция SmartState, которая помечает в реестре образы контейнеров с уязвимостями, после чего OpenShift может блокировать их запуск.

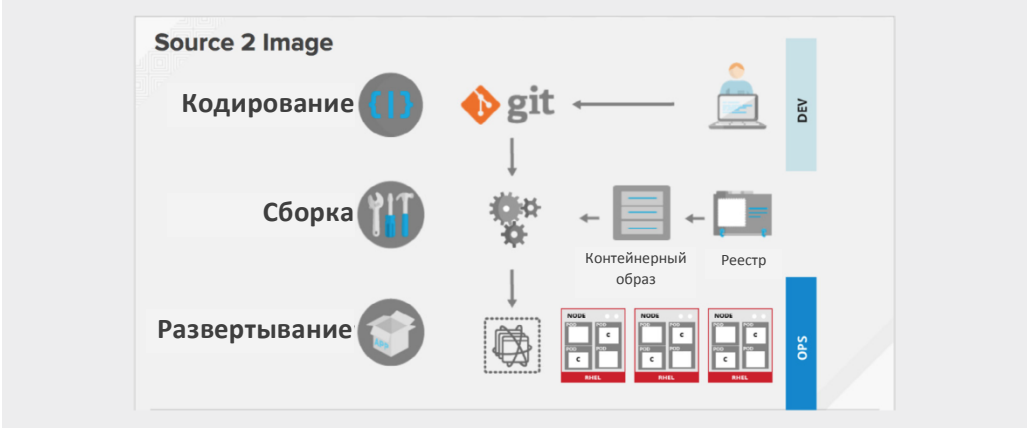
Ниже мы подробнее остановимся на том, как OpenShift поддерживает автоматизацию политик безопасности содержимого контейнеров.

4. Безопасность в процессе сборки

В контейнерной среде процесс сборки является одним из этапов жизненного цикла приложения, на котором исходный код интегрируется с необходимыми runtime-библиотеками времени выполнения. Контроль на этапе сборки – это ключ к безопасности программного стека. Следование философии «собираем один раз, развертываем везде» гарантирует, что продукт, полученный на этапе сборки, полностью идентичен тому, что используется в производственной среде. Кроме того, важно придерживаться принципа неизменности контейнеров: работающие контейнеры не надо патчить, их следует пересобирать и развертывать заново.

Red Hat OpenShift предлагает целый ряд средств обеспечения безопасности и управления на этапе сборки:

- Source-to-image (S2I) – открытый фреймворк для объединения исходного кода ПО и базовых контейнерных образов, облегчающий сотрудничество разработчиков и операторов ИТ-систем при сборке контейнеризованных приложений. Благодаря S2I, OpenShift может автоматически выполнять следующие действия после фиксации изменений исходного кода в git:
    - Запускать (посредством перехватчика webhook в репозитории кода или какого-то другого автоматизированного CI-процесса) автоматическую сборку нового образа из доступных компонентов, включая базовый образ S2I и свежеизмененный код;
    - Автоматически развертывать новую сборку для тестирования;
    - Переводить оттестированный образ в разряд готовых и автоматически развертывать его в производственной среде в рамках CI-процесса.



Red Hat OpenShift содержит интегрированный экземпляр CI-системы Jenkins, а также предлагает RESTful API для интеграции с другими средствами сборки, CI-инструментами или частными реестрами образов, наподобие JFrog Artifactory.

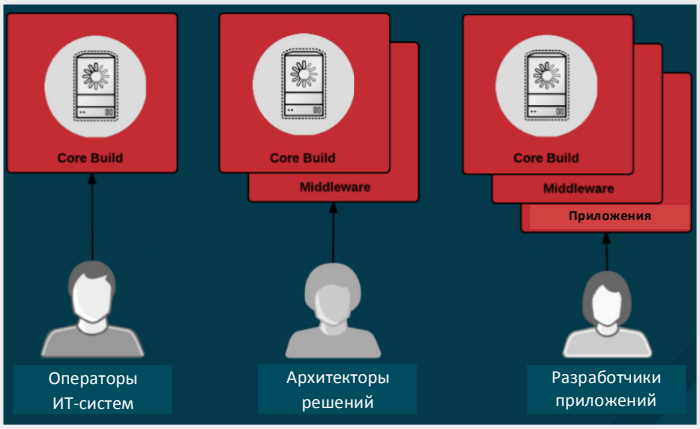
Лучший способ обеспечить безопасность приложений – сделать автоматизированное тестирование безопасности частью процесса сборки или непрерывной интеграции, например, используя следующие инструменты:

- Средства тестирования защищенности SAST (Static Application Security Testing) и DAST (Dynamic Applications Security Testing), такие как HP Fortify и IBM App;
- Сканеры безопасности, выполняющие поиск известных уязвимостей в реальном времени, наподобие Black Duck Hub или JFrog Xray. Такие сканеры каталогизируют содержащиеся в контейнерах пакеты с открытым кодом и регулярно проверяют их по обновляемой базе известных уязвимостей.

Кроме того, процесс непрерывной интеграции должен помечать сборки, имеющие проблемы по результатам тестирования безопасности, чтобы вы могли оперативно принимать соответствующие меры.

Неважно, обязывают ли вас к этому регулятивные требования, или же вы просто хотите оптимизировать командную работы, мы советуем распределить ответственность за сборку и управление образам, исходя из многослойной природы контейнера:

- Операторы ИТ-систем управляют базовыми образами (Core Build).
- Архитекторы решений – связующим ПО (Middleware), компонентами времени выполнения, БД и другими подобными вещами.
- Разработчики концентрируются на слое приложения и просто пишут код.



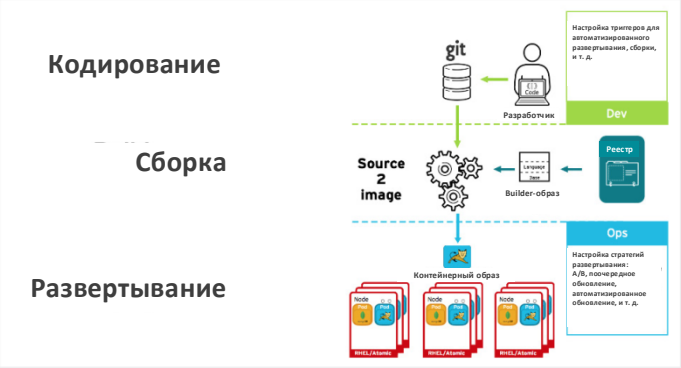
И наконец, мы рекомендуем подписывать контейнеры, которые вы создаете, электронной подписью, чтобы исключить подмену в промежутке между сборкой и развертыванием.

5. Развертывание: контролируем, что может попасть в кластер

Если проверка безопасности при сборке дала сбой, или уязвимость появилась уже после развертывания образа, в дело вступает следующий уровень защиты – инструменты автоматизированного развертывания на основе политик.

Рассмотрим приложение, контейнерный образ которого содержит три слоя: базовый, middleware и слой приложения. В базовом образе обнаруживается уязвимость, после чего он собирается заново и отправляется в реестр OpenShift. OpenShift видит, что базовый образ изменился, и для сборок, которые зависят от этого образа и имеют соответствующие триггеры, автоматически пересобирает образ приложения, включая в него исправленные библиотеки.

По завершении сборки образ приложения вносится во внутренний реестр OpenShift, OpenShift немедленно обнаруживает, что образ во внутреннем реестре изменился, и автоматически развертывает новый образ для приложений, у которых определены соответствующие триггеры. В результате, код, выполняемый в промышленной среде, всегда идентичен коду в последней версии образа, а безопасность становится интегрированной частью процессов CI/CD.



Для усиления защиты также пригодятся политики, позволяющие управлять развертыванием контейнеров с позиций безопасности. Например, OpenShift имеет встроенный механизм ограничения контекста безопасности SCC (Security Context Constraints), с помощью которого можно задать набор условий, которым должен соответствовать pod (квант управления контейнерной платформы, содержащий один или несколько контейнеров), чтобы быть допущенным к развертыванию. Реализованный в OpenShift механизм **SCC**, вошедший в состав Kubernetes под названием Pod Security Policy, позволяет администратору контролировать следующие вещи:

- Запуск **привилегированных контейнеров**;
- Полномочия, которые контейнер может запрашивать;
- Использование каталогов файловой системы хоста в качестве дисковых томов;
- SELinux-контекст контейнера;
- Создание **профилей seccomp**;
- User ID контейнера.

При необходимости и наличии соответствующих полномочий пользователи могут ослабить ограничения политик SCC по умолчанию. И, как и в случае с CI-инструментами, у вас есть возможность интегрировать контейнерную платформу с уже используемыми средствами непрерывного развертывания (CD).

Интеграция инструментов CI/CD с OpenShift позволяет полностью автоматизировать процесс пересборки приложений для включения последних исправлений и тестирования и организовать их развертывание во всех используемых средах.

6. Оркестрация контейнеров: защищаем контейнерную платформу

Приложения редко поставляются в виде одного контейнера. Даже самые простые из них обычно имеют внешний интерфейс, серверную часть и БД. Тем более, современные приложения на основе микросервисов, развертывание которых приводит к появлению множества контейнеров, иногда на одном узле (node), а иногда и сразу на нескольких, как показано на схеме ниже.

При развертывании контейнеров в массовых масштабах надо учесть и продумать следующие моменты:

- На каких хостах будут развертываться те или иные контейнеры;
- Какие хосты имеют большую мощность;
- Какие контейнеры должны взаимодействовать между собой, и как они будут находить друг друга;
- Как контролировать и управлять доступом к общим ресурсам, наподобие сети или хранилищ;
- Как мониторить исправность контейнеров;
- Как организовать автомасштабирование приложений;
- Как предоставить разработчикам возможности самообслуживания без ущерба для безопасности.

Инструменты управления контейнерами можно спроектировать и построить самостоятельно. Но зачем тратить время, если есть готовая контейнерная платформа с встроенными средствами управления и безопасности?

Red Hat OpenShift Container Platform реализует оркестрацию контейнеров, автоматизацию планирования задач и запуск контейнеров приложений в кластерах на основе физических и виртуальных машин, используя и расширяя возможностей открытой технологии Kubernetes. Kubernetes – это проект с открытым кодом, начатый компанией Google и решающий проблему оркестрирования контейнерными кластерами за счет использованием управляющих серверов (master) и управляемых узлов (node). Платформа Red Hat также включает в себя систему управления облачными средами CloudForms, которая, среди прочего, обеспечивает мониторинг исправности контейнеров в вашем частном реестре и предотвращает развертывание контейнеров со свеженайденными уязвимостями.

Учитывая богатство возможностей, которые контейнерная платформа предоставляет разработчикам и ИТ-специалистам, здесь просто необходим строгий контроль доступа на основе ролей. Например, управляющие серверы оркестрации являются отправной точкой при доступе к системе и должны иметь самый высокий уровень безопасности. В свою очередь, API – это ключ к автоматизации массового управления контейнерами. Они применяются для проверки и настройки данных для pod‘ов, сервисов и контроллеров репликации, для проверки проекта по входящим запросам и активации триггеров в других ключевых компонентам системы.

Поэтому контроль доступа к API (аутентификация и авторизация) – это тоже ключевой элемент безопасности контейнерной платформы. **Управляющие серверы** OpenShift имеют встроенный **сервер OAuth**, а разработчики и администраторы получают **маркеры доступа OAuth**, чтобы удостоверять себя при обращении к API. Администратор может настроить проверку подлинности OAuth с использованием сторонних поставщиков удостоверений, включая каталоги LDAP.

Еще одно преимущество контейнерной платформы – возможности самообслуживания для разработчиков, помогающие быстрее и проще собирать приложения на основе контейнерных слоев, созданных другими специалистами. Как следствие, платформа должна обеспечивать безопасность в мультитенантной среде, чтобы команды разработки не могли получить доступ к средам друг друга без авторизации, а портал самообслуживания позволял им гибко управлять доступом при совместной работе без ущерба для безопасности. OpenShift расширяет возможности Kubernetes в плане обеспечения безопасности мультитенантного главного хоста, в результате чего:

- Все обращения к управляющим серверам выполняются по протоколу TLS.
- Обращения к серверу API осуществляются только с использованием сертификатов X.509 или токенов доступа.
- Использование квот в проекте позволяют ограничить возможный ущерб от незаконных токенов.
- Распределенное хранилище параметров конфигурации Etcd не предоставляется напрямую кластеру.

OpenShift также предлагает **расширенные средства управления кластером**, включая улучшенные возможности управления секретами и сертификатами.

7. Изоляция на уровне сети

Контейнерное развертывание приложений на основе микросервисов зачастую приводит к появлению большого числа контейнеров, распределенных по нескольким узлам (nodes) кластера. Приложения в таком кластере надо как-то изолировать друг от друга, хотя бы из общих соображений безопасности.

Ведущие облачные службы контейнеров, наподобие Google Container Engine (GKE), Azure Container Services или Amazon Web Services (AWS) Container Service, не рассчитаны на мультитенантость. Они лишь позволяют запускать контейнеры в кластере ВМ, который вы там создаете. Поэтому для обеспечения безопасности в режиме мультитенантости понадобится контейнерная платформа, сегментирующая сетевой трафик в кластере с целью изоляции пользователей, групп, приложений и сред.

Сетевые пространства имен позволяют привязать каждый pod на узле к собственному диапазону IP-адресов и портов, обеспечивая изоляцию на уровне сети. По умолчанию pod'у из одного пространства имен (проекта) запрещено обмениваться пакетами с pod'ами и сервисами из других проектов (ниже мы покажем, как это разрешить). Таким образом, можно легко изолировать друг от друга среду разработки, среду тестирования и промышленную среду, которые находятся в одном кластере.

Однако разведение pod'ов по диапазонам IP-адресов и портов усложняет сеть. Да и природа контейнеров такова, что они постоянно возникают и исчезают. Поэтому мы советуем вложиться в инструменты, которые будут решать возникающие здесь проблемы за вас. Желательно, в контейнерные платформы, которые используют программно-определяемые сети SDN и предоставляют унифицированную сеть кластера с возможностью коммуникации контейнеров друг с другом.

Также стоит отдавать предпочтение платформам, позволяющим контролировать входящий (Ingress) трафик с помощью **маршрутизатора** или **брандмауэра**, чтобы, например, разрешать доступ к БД только с заданных IP-адресов.

SDN также предоставляет еще один уровень защиты, вдобавок к сетевым пространствам имен. Для этого используется плагин **ovs-multitenant**, который изолирует трафик на уровне проектов (пространств имен), где проект – это набор pod'ов и сервисов с одинаковым идентификатором виртуальной сети. По умолчанию pod'ы из разных проектов не могут обмениваться пакетами. Но в OpenShift Container Platform 3.1 появилась функция **oadm pod-network**, с помощью которой можно разрешить выбранным проектам обращаться к сервисам друг друга, или же разрешить всем проектам доступ ко всем pod'ам и сервисам в кластере. Эта функция работает на уровне проекта в целом и всегда разрешает трафик в обе стороны. Иначе говоря, если вы имеете доступ к одному сервису в каком-то проекте, то имеете доступ и ко всем остальным сервисам в этом проекте, а также – внимание! – *этот проект имеет доступ ко всем сервисам в вашем проекте*. В силу двусторонности выдавать такие разрешения может только администратор кластера.

В Red Hat OpenShift Container Platform 3.7 появилась **версия** нового плагина Network Policy (ovs-networkpolicy), который расширяет возможности настройки разрешений для трафика между pod'ами по сравнению с ovs-multitenant, позволяя конфигурировать политики изоляции на уровне отдельных pod'ов. И поскольку разрешения Network Policy не являются двусторонним и применяются к входящему трафику только тех pod'ов, которые находятся под контролем администратора проекта, для их использования не нужны полномочия администратора кластера.

Сетевые сканеры тоже можно развертывать и запускать в виде контейнеров с помощью, так называемых, **контейнеров с супер привилегиями**.

8. Хранилища

Контейнеры полезны как для приложений stateless (без отслеживания состояний), так и для приложений stateful (с отслеживанием состояний). Защита хранилища – ключевой элемент безопасности stateful-сервисов. Red Hat OpenShift Container Platform имеет плагины для подключения различных **хранилищ**, включая **NFS**, **AWS Elastic Block Stores (EBS)**, **GCE Persistent Disks**, **GlusterFS**, **iSCSI**, **RADOS (Ceph)** и Cinder.

Постоянные тома PV (Persistent Volume) можно подключать на хосте любым способом, который поддерживается поставщиком ресурсов хранения данных. Например, NFS поддерживает режимы ReadWriteOnce (только один клиент, чтение-запись), ReadOnlyMany (несколько клиентов, только чтение) и ReadWriteMany (несколько клиентов, чтение-запись). Но конкретный PV-том NFS можно экспортировать на сервер как read only.

Для сетевых файловых систем (NFS, Ceph, Gluster, и т. д.) трюк заключается в том, что PV-том регистрирует свой идентификатор GID (group ID) в качестве аннотации на ресурсе PV. Поэтому когда pod запрашивает PV, этот аннотированный GID добавляется в **supplemental-группы** pod'а, предоставляя pod'у доступ к содержимому общего хранилища.

При работе с блочными хранилищами (EBS, GCE Persistent Disks, iSCSI, и т. д) контейнерная платформа может применять средства SELinux для защиты корневого объекта подключенного тома для непривилегированных pod'ов , чтобы этот том принадлежал и был виден только тому контейнеру, с которым он ассоциирован.

Весь обмен данными между компонентами контейнерной платформы нужно защищать с помощью HTTPS. И, конечно же, следует пользоваться средствами безопасности имеющейся СХД.

9. Управление API, защита конечных точек сервисов и SSO

Безопасность включает в себя аутентификацию и авторизацию приложений и API. Современным приложениям, как правило, требуется служба единого входа на веб-сайты (Web SSO), такая как, например, Red Hat SSO (RH-SSO), входящая в состав нашей контейнерной платформы. RH-SSO предлагает готовую к использованию систему аутентификации, которая поддерживает стандарты SAML 2.0 и OpenID Connect, реализует функционал Web SSO и содержит службу федерации на основе открытого проекта Keycloak.

RH-SSO 7.1 предлагает клиентские адаптеры для Red Hat JBoss Fuse и Red Hat JBoss Enterprise Application Platform (JBoss EAP), а также новый клиентский адаптер, обеспечивающий аутентификацию и механизмы Web SSO для приложений Node.js. RH-SSO поддерживает интеграцию со службами каталогов LDAP, включая Microsoft Active Directory и Red Hat Enterprise Linux, а также позволяет использовать для входа аккаунты Facebook, Google и Twitter.

Для приложений, скомпонованных из микросервисов, трудно переоценить значение API. Такие приложения имеют несколько независимых API-сервисов, что ведет к разрастанию конечных точек, которые требуется контролировать, и, как следствие, к необходимости использовать специальные инструменты регулирования. Кроме того, мы советуем применять системы управления API-интерфейсами, такие как 3Scale от Red Hat. Эта система поддерживает различные стандарты аутентификации и защиты API (включая API-ключи, пары ключей и идентификаторы приложений, а также OAuth 2.0) и позволяет использовать их вместе или по отдельности при выдаче удостоверений и контроле доступа.

Возможности 3Scale по управлению доступом не ограничиваются базовым функциями защиты и аутентификации. Решение может ограничивать доступ к отдельным конечным точкам, методам и сервисам с использованием карт (plans) приложений и учетных записей, а также создавать и применять политики доступа для пользователей и групп. Карты приложений позволяют ограничить интенсивность обращений к API и контролировать количество вызовов на уровне отдельных групп разработчика. Лимиты на входящие API-вызовы в заданном интервале времени позволяют защитить инфраструктуру и сгладить пики трафика, а настраиваемые триггеры, срабатывающие при достижении приложением пороговых значений – уведомить администраторов и выполнить другие заданные действия.

10. Управление ролями и доступом в федеративных кластерах

В выпущенной в июле 2016 года Kubernetes 1.3 были впервые представлены федеративные кластеры, которые на момент Kubernetes 1.6 находятся в стадии бета. Эти кластеры значительно облегчают развертывание и организацию доступа к сервисам приложений в мультикластерной среда на базе общедоступного облака или корпоративных дата-центров. Мультикластерные конфигурации могут применяться для обеспечения высокой доступности приложений при наличии нескольких ЦОД или для управления развертыванием и миграцией в средах на базе облачных платформ различных поставщиков, таких как AWS, Google Cloud и Azure.

С появлением федеративных кластеров возникает потребность в средствах оркестрации, способных обеспечить безопасность в условиях многообразия платформ развертывания. И ключом к решению этой задачи по-прежнему является аутентификация и авторизация, а также возможность организовать защищенную передачу данных приложениям вне зависимости от того, где они запускаются, и эффективно поддерживать мультитенантность приложений в среде с несколькими кластерами. Участники проекта Kubernetes активно работают над расширением возможностей федеративных кластеров за счет федеративных секретов (Federated Secrets), пространств имен (Federated Namespaces) и объектов Ingress.

Федеративные секреты – механизм автоматического создания и сопровождения секретов на всех кластерах федерации для обеспечения целостности и актуальности даже при недоступности отдельных кластеров на момент запуска обновлений.

Федеративные пространства имен – примерно то же, что традиционные **пространства имен Kubernetes**, но создаются на уровне федерации и синхронизируются на всех ее кластерах.

Red Hat тесно работает с сообществом разработки Kubernetes и введет эти новые возможности в состав платформы OpenShift после их окончательной реализации.

ЗАКЛЮЧЕНИЕ

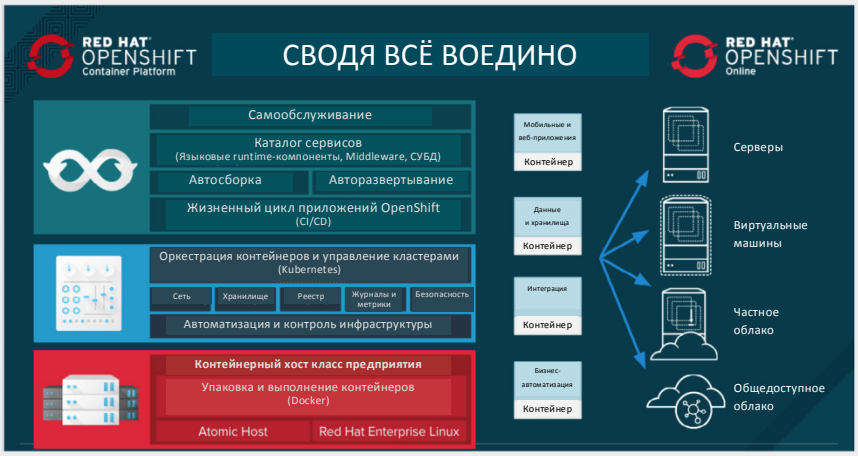
Разумеется, безопасность – это отнюдь не все. Платформа контейнеризации приложений должна улучшать работу разработчиков и операторов ИТ-систем. Это должно быть защищенное решение корпоративного класса, которое предоставляет и тем, и другим весь необходимый функционал, повышает операционную эффективность и КПД ИТ-инфраструктуры.

Платформа Red Hat OpenShift 3, в основе которой лежат отраслевые стандарты и переносимые Linux-контейнеры, предлагает следующие преимущества в плане обеспечения безопасности:

- Строгое управление доступом на основе ролей с возможностью подключения корпоративных систем аутентификации.
- Мощные средства массовой оркестрации и управления контейнерами на основе технологии Google Kubernetes.
- Интегрированные ОС Red Hat Enterprise Linux 7 и Red Hat Enterprise Linux Atomic Host, оптимизированные для массовой эксплуатации контейнеров с возможностью усиленной изоляции средствами SELinux.
- Интеграция с общедоступными и частными реестрами.
- Интегрированные инструменты CI/CD для реализации методологии DevOps.
- Новая сетевая модель контейнеров.
- Поддержка удаленных томов хранения.

OpenShift уже поддерживает крупнейший набор языков программирования, фреймворков и сервисов, и вскоре получит функционал федеративных кластеров.

OpenShift может работать на платформах OpenStack, VMware, AWS, GCP, Azure и везде, где есть Red Hat Enterprise Linux 7. Решение также предлагается в виде общедоступных облачных услуг OpenShift Dedicated и OpenShift Online.



Имея за плечами более 15 лет успешного опыта производства и поставки надежных и проверенных решений корпоративного класса с полностью открытым кодом, Red Hat сегодня задает новый стандарт доверия и безопасности в сфере контейнеризации, предлагая Red Hat Enterprise Linux, Red Hat OpenShift Container Platform и целый ряд своих продуктов и решений в виде контейнеров.

О КОМПАНИИ RED HAT

Red Hat – это ведущий поставщик надежных и высокопроизводительных технологий облачных вычислений, виртуализации, хранения данных, связующего ПО и операционных систем Linux, в основе которых лежат решения с открытым кодом, развиваемые силами сообщества разработчиков. Компания также предлагает неоднократно отмеченные наградами услуги технической поддержки, обучения и консалтинга. Выступая в качестве центрального узла всемирной сети корпоративных заказчиков, партнеров и сообществ разработки открытого ПО, Red Hat способствует созданию инновационных технологий, раскрепощающих ресурсы для роста и помогающих во всеоружии встретить будущее.

СЕВЕРНАЯ АМЕРИКА
1 888 REDHAT1

ЕВРОПА, БЛИЖНИЙ ВОСТОК И АФРИКА
00800 7334 2835
europe@redhat.com

АЗИАТСКО-ТИХООКЕАНСКИЙ РЕГИОН
+65 6490 4200
apac@redhat.com

ЛАТИНСКАЯ АМЕРИКА
+54 11 4329 7300
info-latam@redhat.com

facebook.com/redhatinc
@redhatnews
linkedin.com/company/red-hat

redhat.com
#f7530_0517