Teach Me Skills

Защита инфраструктуры приложений

Teach Me Skills

Вопросы по предыдущим темам или ДЗ

Mini-quize по прошлым темам:

1. **В какое место сетевой схемы устанавливают IDS/IPS?**
2. **Что такое Windows Defender?**
3. **Как можно защититься от DDOS?**
4. **Как можно защитить почтовый сервер?**
5. **Что означает CVSS?**
6. **Что такое Ip tables и чем он может быть полезен?**
7. **Что такое Zero Trust?**

Mini-quize по новой теме:

1. **Что такое виртуализация?**
2. **Что означает термин отказоустойчивый сервер?**
3. **Как осуществляется безопасность облачных сред?**
4. **Есть ли категорирование уязвимостей для Docker, как вы думаете она называется?**
5. **Что такое docker compose, можно ли создать собственный?**

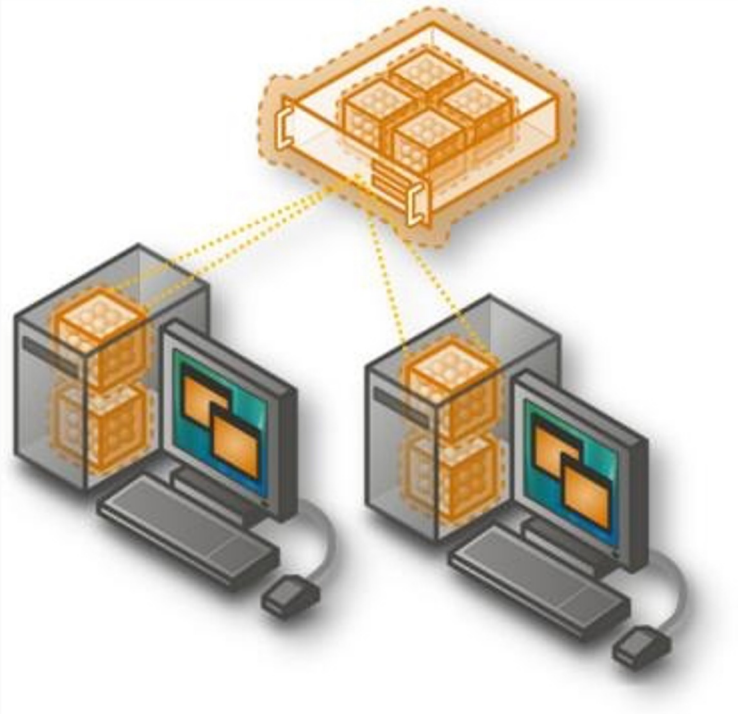
Teach Me Skills

План занятия

1. Виртуализация и отказоустойчивость
2. Как обеспечивается безопасность в AWS, GCP, Яндекс облаке и тд
3. Docker, оркестрация и методы их защиты

Защита инфраструктуры приложений

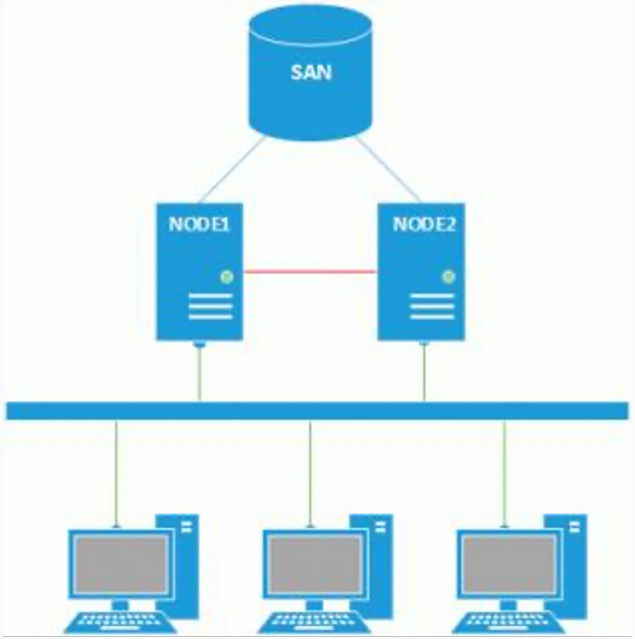
Виртуализация



Защита инфраструктуры приложений

Отказоустойчивый - с обработкой отказа, хотя сегодня все чаще употребляется иной термин - высокой доступности (high availability)

По схеме работы узлы могут работать в режиме **активный-пассивный** или **активный-активный**. В первом случае все клиентские запросы обслуживаются одним из узлов, второй узел вступает в работу только при отказе первого. Второй вариант предусматривает обработку клиентских запросов обоими узлами, при этом также можно осуществлять балансировку нагрузки и увеличивать вычислительные ресурсы, путем добавления новых узлов кластера. В случае отказа одного из узлов клиентские запросы обрабатывают оставшиеся ноды.



Защита инфраструктуры приложений

Виртуализация и отказоустойчивость

Защита инфраструктуры приложений

Защита виртуальных серверов

Kaspersky Security для виртуальных и облачных сред

Специализированная защита виртуальных и облачных сред

Kaspersky

Security для виртуальных сред Легкий агент

Требования

Обеспечение безопасности в Облачных инфраструктурах

Облачная безопасность предоставляет организациям подход к решению проблем безопасности и обеспечивает соблюдение нормативных требований. Эффективное обеспечение безопасности облака требует наличия нескольких уровней защиты в рамках стека облачных технологий, в который входят следующие:

- **Средства превентивного контроля**, предназначенные для блокировки авторизованного доступа к конфиденциальным системам и данным.
- **Средства детективного контроля**, предназначенные для выявления несанкционированного доступа к системам и данным и их изменений посредством проведения аудита, мониторинга и отчетности.
- **Средства автоматического контроля**, предназначенные для предотвращения, обнаружения и реагирования на обновления безопасности, как регулярные, так и критически важные.
- **Средства административного контроля**, разработанные для контроля за применением политик, стандартов, практик и процедур безопасности.

Машинное обучение и искусственный интеллект позволяют дополнить портфель систем облачной безопасности технологиями контекстной осведомленности.

Защита инфраструктуры приложений

Какие еще требования важны для обеспечения безопасности облачных данных?

Общая ответственность за безопасность и доверие: доверие имеет первостепенное значение при выборе партнера по облаку, который будет отвечать за свою часть модели общей безопасности. Организации должны четко понимать свои роли и обязанности, а также иметь доступ к независимым сторонним аудитам и аттестациям систем безопасности.

Автоматизация и машинное обучение: облачные угрозы несутся со скоростью автомобиля, а традиционные корпоративные системы безопасности могут анализировать инциденты и реагировать на них со скоростью человека. Современная система безопасности в облачных средах должна автоматизировать обнаружение угроз и реагирование на них. Для сложных угроз требуются новые современные решения безопасности, которые способны прогнозировать, предотвращать, обнаруживать угрозы и и реагировать на них с помощью машинного обучения.

Эшелонированная защита: многоуровневая система безопасности по всему технологическому стеку должна включать средства превентивного, детективного и административного контроля за соответствующими людьми, процессами и технологиями для обеспечения безопасности физических центров обработки данных поставщиков облачных услуг.

Какие еще требования важны для обеспечения безопасности облачных данных?

Управление учетными записями: по мере того как мобильные устройства, приложения и сведения о пользователях используются все шире, учетные записи становятся новым периметром. Важнейшее значение имеет контроль доступа и привилегий в облаке и локальных системах.

Прозрачность: брокер защиты доступа в облако и решение для управления средствами обеспечения безопасности в облаке увеличивают прозрачность и контроль над всей облачной средой организации.

Постоянное соответствие требованиям: соответствие нормативным требованиям является обязательным, но соответствие и безопасность – это не одно и то же. Организации могут нарушить нормативные требования, не нарушая при этом безопасность, например, в результате изменений и ошибок конфигурации. Для компаний очень важно иметь решение по управлению облачными средами, которое предоставляет полные, своевременные и действенные данные, связанные с соблюдением нормативных требований, по всем облачным средам.

Защита инфраструктуры приложений

Какие еще требования важны для обеспечения безопасности облачных данных?

Безопасность по умолчанию: поставщик облачных услуг должен активировать средства контроля безопасности по умолчанию, а не требовать от предприятия помнить о том, что их нужно включать. Не все имеют четкое представление о различных средствах управления безопасностью и о том, как они работают вместе для снижения риска и создания полноценной системы безопасности. Например, шифрование данных должно быть включено по умолчанию. В облаках должны применяться согласованные средства контроля и политики защиты данных.

Мониторинг и миграция: для обеспечения безопасности рабочих нагрузок администраторам политик безопасности следует настроить и обеспечить соблюдение политик безопасности для облачных пользователей и секций. Унифицирование представление всех средств контроля облачной безопасности по всем пользователям облака также необходимо для обнаружения ошибок конфигурации ресурсов и небезопасных действий по всем пользователям, что предоставляет администраторам безопасности возможность отслеживать и решать проблемы безопасности облака.

Разделение обязанностей и доступ с минимальными привилегиями: принципы разделения обязанностей и доступа с минимальными привилегиями – это практические рекомендации по безопасности, которые следует применять в облачных средах. Таким образом Вы сможете гарантировать, что отдельные лица не будут обладать чрезмерными административными правами и не смогут получить доступ к конфиденциальным данным без дополнительной авторизации.

МНОГОУРОВНЕВАЯ ЗАЩИТА БЕЗОПАСНОСТИ ОБЛАЧНОЙ ИНФРАСТРУКТУРЫ

Один из способов обеспечения безопасности облачной инфраструктуры — создание многоуровневой защиты. В ее состав входят следующие элементы:

- физическая защита дата-центров, доступ к серверам, системам охлаждения, электропитанию и прочим физическим ресурсам;
- инфраструктурная безопасность: применение сетевых брандмауэров, систем обнаружения вторжений, виртуальных частных сетей (VPN) и иных механизмов;
- управление доступом и идентификация, управление привилегиями, ролевая модель доступа и другие механизмы, чтобы обеспечить авторизованный доступ к данным и ресурсам;
- информационная безопасность облачных технологий, к которой относятся методы, обеспечивающие защиту данных от третьих лиц: шифрование, предотвращение утечки и т. д;
- мониторинг и анализ, позволяющие обнаруживать подозрительную активность, атаки или нарушения;
- управление рисками и соответствие требованиям: оценка рисков, разработка политик безопасности, проведение аудитов и соблюдение соответствующих нормативных требований.

Защита инфраструктуры приложений

Защита информации в облачных сервисах

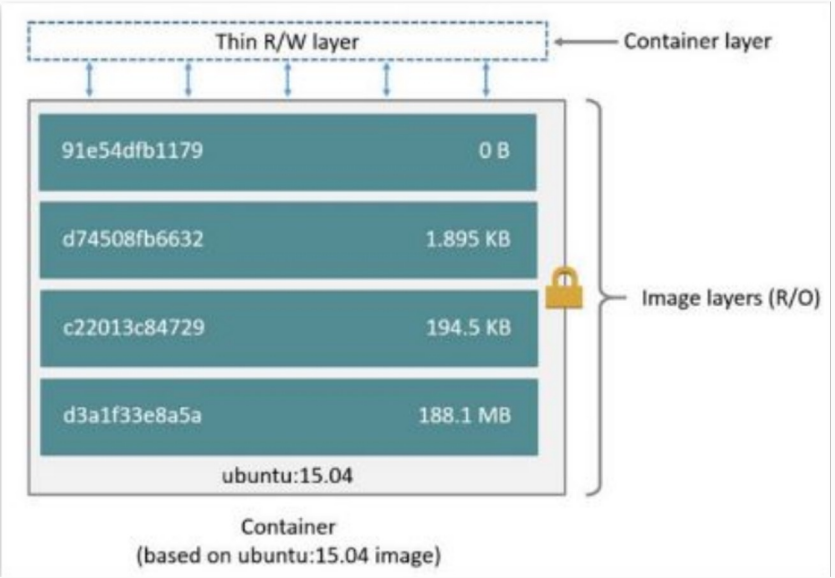
Обеспечение безопасности с Yandex Cloud

Обеспечение безопасности развертывания Azure

Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты
- Работа с данными

Docker использует драйверы хранилища (Storage drivers) для хранения слоев изображений и хранения данных в доступном для записи уровне контейнера



Docker, оркестрация и методы их защиты

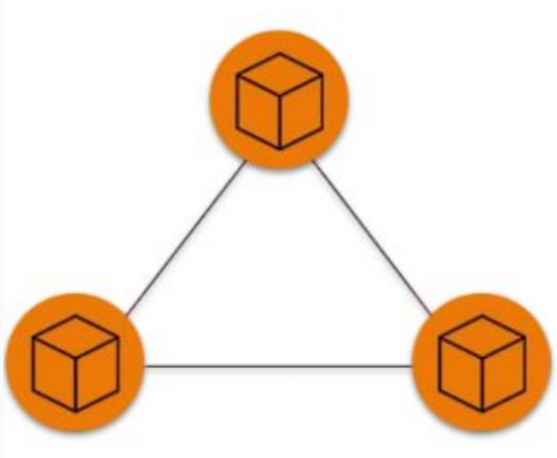
- Работа с данными (типы)

Block level

Блочное хранилище разделяет данные на блоки и хранит их как отдельные части

Блочное хранилище не зависит от единственного пути к данным - в отличие от файлового хранилища его можно получить быстро

Пример блочных storage drivers: DeviceMapper, ZFS



Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты

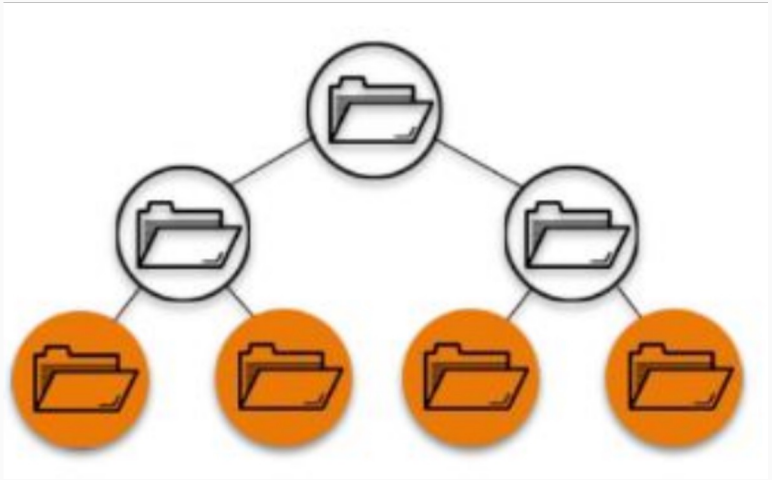
- Работа с данными (типы)

File level

Файловое хранилище, также называемое хранилищем на уровне - это именно то, что вы думаете: данные хранятся в виде единого фрагмента информации внутри директории.

Это самая старая и наиболее широко используемая система хранения данных для систем хранения с прямым подключением к сети.

Пример файловых storage drivers: AUFS, Overlay, Overlay2



Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты

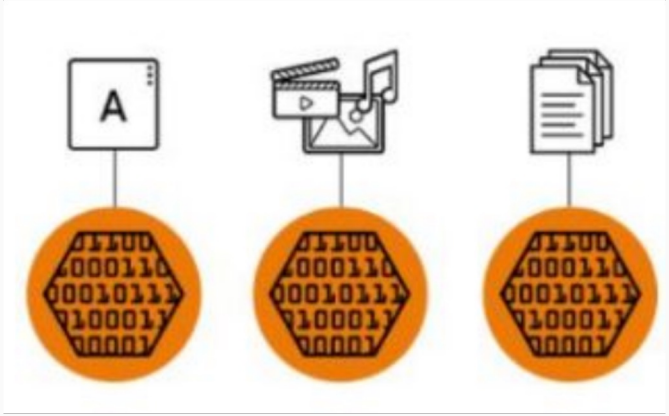
- Работа с данными (типы)

Object level

Объектное хранилище, также известное как объектно-ориентированное хранилище, представляет собой плоскую структуру, в которой файлы разбиваются на части и распределяются по оборудованию:

- для объектного хранилища требуется простой интерфейс прикладного программирования (API)
- объекты не могут быть изменены

Примеры ООХ: Ceph, OpenIO



Docker, оркестрация и методы их защиты

- Работа с данными

Форматы хранилищ

- Файловое хранилище

организует и представляет данные в виде иерархии файлов в папках

- Блочное хранилище

разбивает данные на произвольно организованные тома одинакового размера

- Объектное хранилище

управляет данными и связывает их со связанными метаданными (не рассматривается в Docker)

Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты
- Работа с данными

Форматы хранилищ

Storage Drivers	Type
AUFS (Advanced Union Filesystem)	File level
BTRFS	Block level
DeviceMapper	Block level
Overlay	File level
Overlay2	File level
ZFS	Block level

overlay2	overlay2 является предпочтительным драйвером хранилища для всех поддерживаемых в настоящее время дистрибутивов Linux и не требует дополнительной настройки
overlay	Устаревший overlay драйвер использовался для ядер, которые не поддерживали функцию «multiple-lowerdir»
Btrfs, ZFS	В btrfs и zfs драйверах имеются дополнительные опции, такие как создание «снапшотов», но требуют большего обслуживания и настройки
aufs	Aufs Драйвер запоминающего устройства был предпочтительным драйвер для хранения Докер 18.06 и старше, при работе на Ubuntu 14.04 на ядре 3.13 , которая не имела никакой поддержки overlay2. Однако в текущих версиях Ubuntu и Debian теперь есть поддержка overlay2, которая теперь является рекомендуемым драйвером
devicemapper	Был для производственных сред, потому что loopback-lvm, но имел очень низкую производительность. Devicemapper был рекомендованным драйвером хранилища для CentOS и RHEL, поскольку их версия ядра не поддерживала overlay2. Однако в текущих версиях CentOS и RHEL теперь есть поддержка overlay2

Docker, оркестрация и методы их защиты
- Работа с данными

Storage driver Overlay2 является драйвером по умолчанию

Изменить storage driver можно в конфигурации docker
daemon (/etc/docker/daemon.json)

Текущий storage driver можно узнать с помощью команды
"*docker info*"

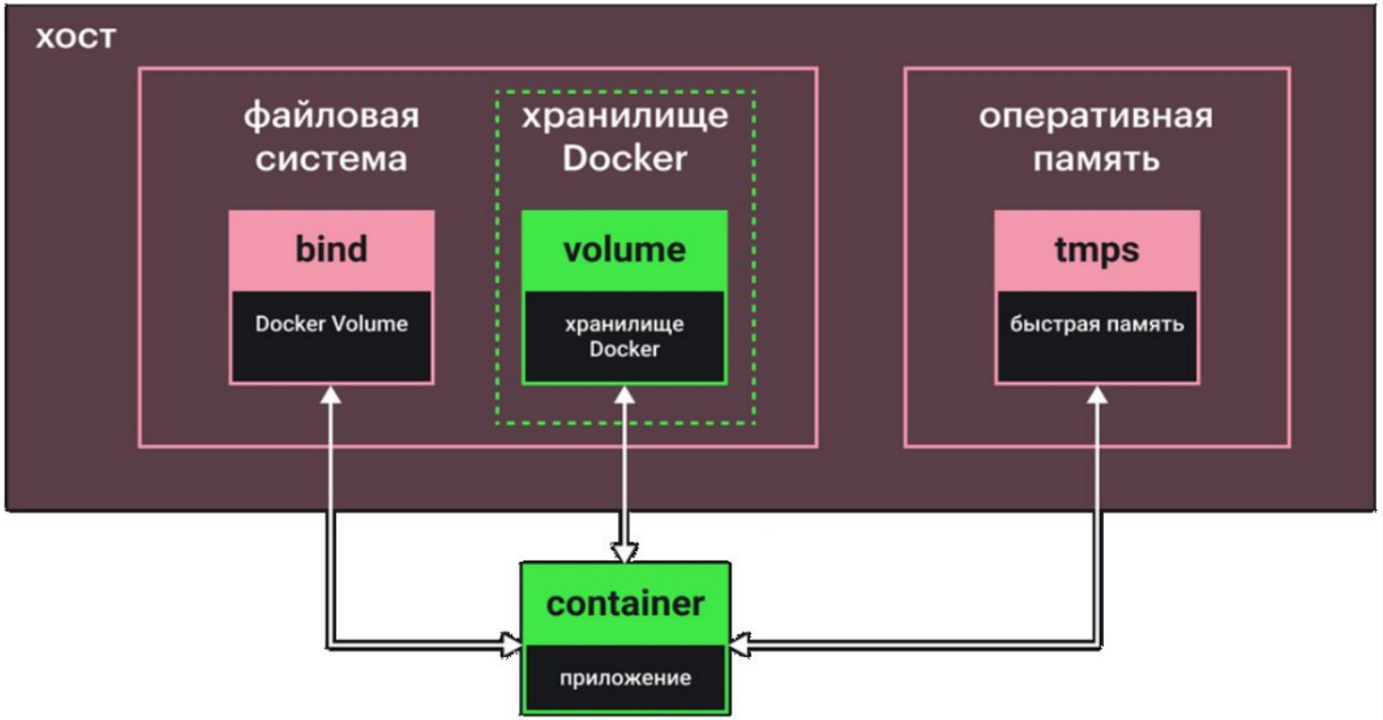
```
Client:
 Version:    24.0.5
 Context:    default
 Debug Mode: false

Server:
 Containers: 0
  Running: 0
  Paused: 0
  Stopped: 0
 Images: 0
 Server Version: 24.0.5
 Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Using metacopy: false
  Native Overlay Diff: true
  userxattr: false
 Logging Driver: json-file
 Cgroup Driver: systemd
 Cgroup Version: 2
 Plugins:
```

Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты
- Работа с данными (docker volumes)

Какие данные при работе контейнера стоит хранить постоянно?

- базы данных
- логи
- файлы конфигураций
- настройки пользователей
- .. и другие



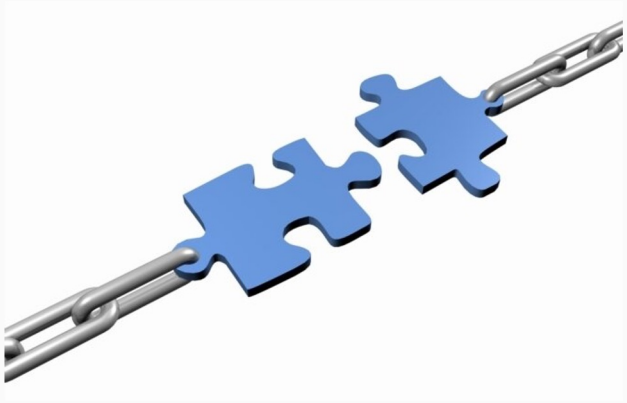
Docker, оркестрация и методы их защиты

- Работа с данными (docker volumes)

Для постоянного хранения данных в docker существует механизм docker volumes, с помощью которых можно:

- выгружать данные контейнера на персистентное хранение
- работать в подмонтированных каталогах с интенсивной записью
- совместно использовать выгружаемые данные сразу в нескольких контейнерах

По умолчанию тома docker volumes создаются в каталоге /var/lib/docker/volumes/



Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты
- Работа с данными (docker volumes)

docker volumes в основном
применяются в docker-compose

```yaml
version: "3"
volumes:
  pgdata:
services:
  pagila:
    image: postgres:13.2
    container_name: pagila
    environment:
      POSTGRES_PASSWORD: 1313
      POSTGRES_USER: postgres
    volumes:
      - ./pagila-schema.sql:/docker-entrypoint-initdb.d/1-pagila-schema.sql
      - ./pagila-data.sql:/docker-entrypoint-initdb.d/2-pagila-data.sql
      - pgdata:/var/lib/postgresql/data
    expose:
      - 5432
    ports:
      - 6543:5432
  pgadmin:
    container_name: pgadmin4_container
    image: dpage/pgadmin4
    restart: always
    ports:
      - 5050:80/tcp
    environment:
      PGADMIN_DEFAULT_EMAIL: admin@admin.com
      PGADMIN_DEFAULT_PASSWORD: root
      PGADMIN_LISTEN_PORT: "5050"
```

Docker, оркестрация и методы их защиты

- Работа с сетями

Network Drivers
- Bridge
- Host
- Overlay
- Macvlan
- None
- Network plugins*

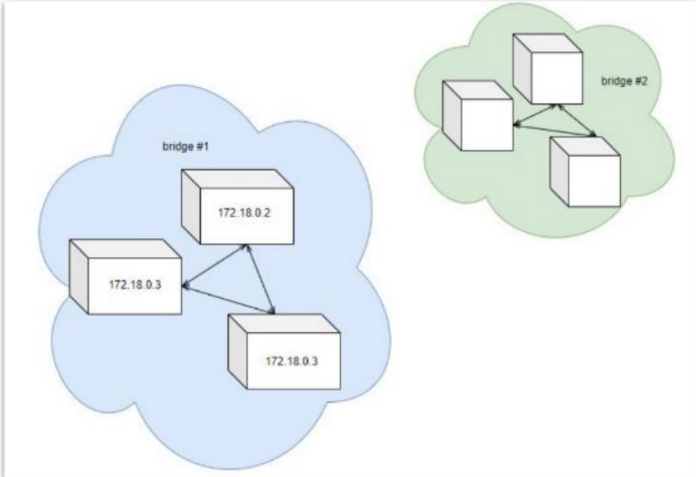
Bridge network driver (by default)
- Контейнеры, объединенные в такую сеть (bridge) могут общаться друг с другом, но не
могут напрямую обратиться к контейнерам из другой bridge-сети
- Является сетевым драйвером по умолчанию

Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты

- Работа с сетями

Network Drivers
- Bridge
- Host
- Overlay
- Macvlan
- None
- Network plugins*



Bridge network driver (by default)

- Контейнеры, объединенные в такую сеть (bridge) могут общаться друг с другом, но не
могут напрямую обратиться к контейнерам из другой bridge-сети
- Является сетевым драйвером по умолчанию

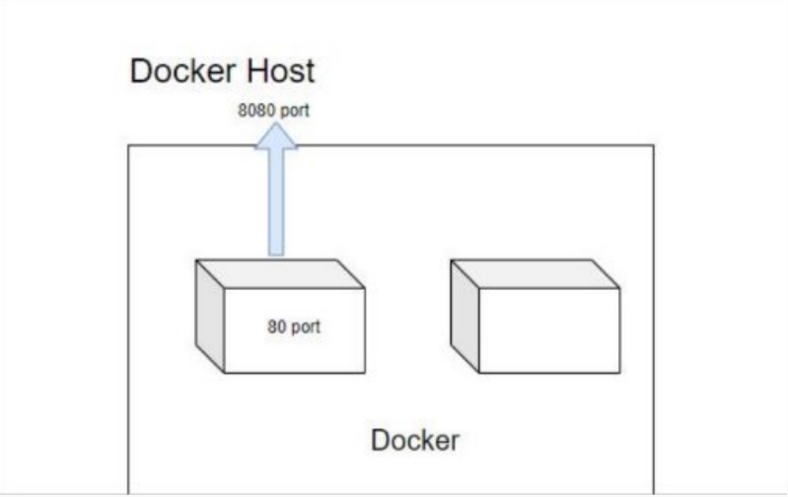
Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты

- Работа с сетями

Host network driver

Драйвер удаляет сетевую изоляцию между контейнером и хостом Docker и напрямую использует сеть хоста



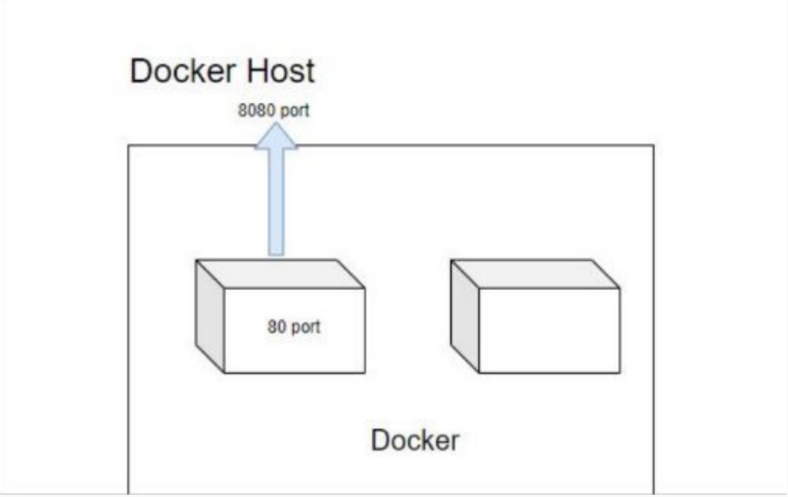
Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты
- Работа с сетями

Host network driver

Драйвер удаляет сетевую изоляцию между контейнером и хостом Docker и напрямую использует сеть хоста

P.s. Работает только на Linux хостах



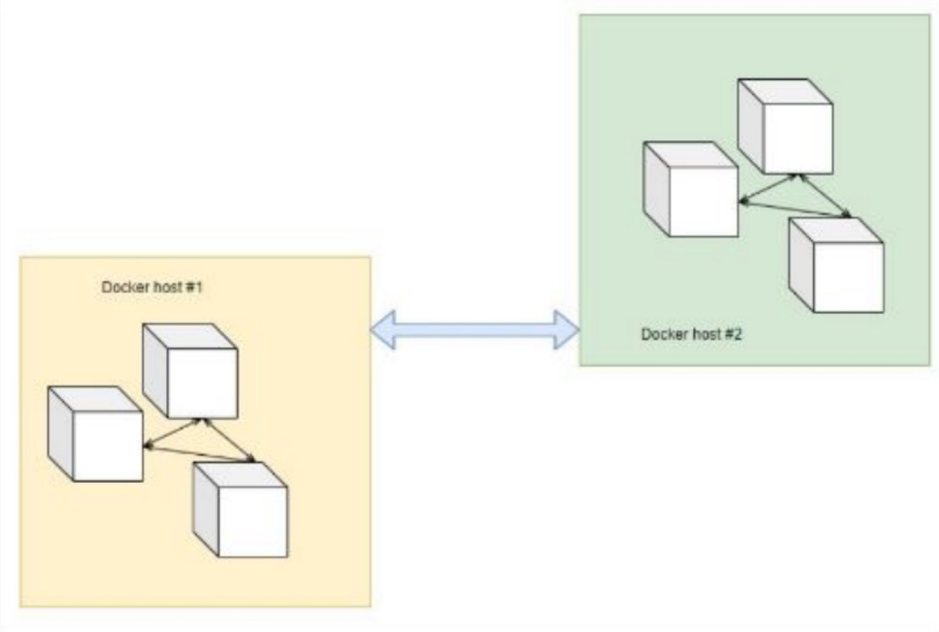
Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты

- Работа с сетями

Overlay network driver
Объединяет сети контейнеров на разных Docker-хостах

P.s. Не поддерживает шифрование на Windows хостах

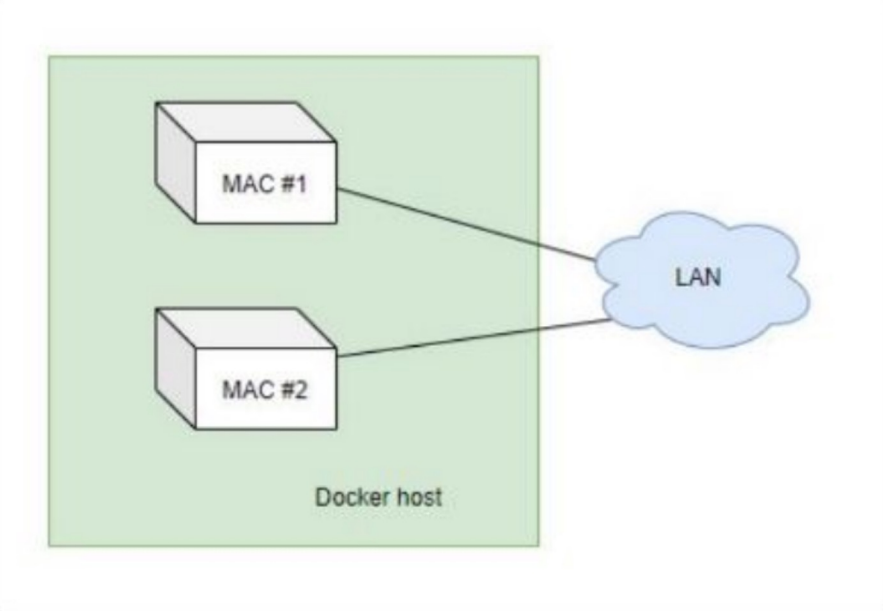


Docker, оркестрация и методы их защиты
- Работа с сетями

Macvlan network driver

Присваивает каждому контейнеру физический MAC-адрес, позволяющий обращаться к нему как к настоящему устройству

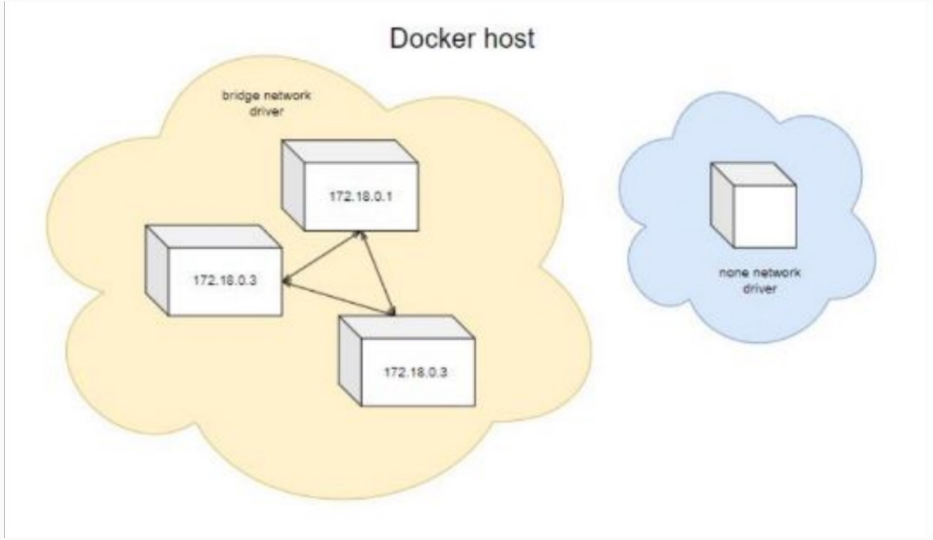
P.s. Работает только на Linux хостах



Docker, оркестрация и методы их защиты
- Работа с сетями

None network driver

Отключает все сети для контейнера, к которому применен



Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты
- Работа с сетями

Network plugins

Подключаемые сторонние сетевые плагины*

Плагин	Описание
Contiv Networking	- Открытый исходный код - Обеспечивает инфраструктуры и политики безопасности для развертывания мультитенантных микросервис - Интеграция с физической сетью для неконтейнерных рабочих нагрузок
Kuryr Network	- Разработан в рамках проекта OpenStack Kuryr - Реализует API удаленного драйвера сети Docker (libnetwork) с помощью Neutron (сетевой службы OpenStac) - Включает драйвер IPAM.
Weave Network	- Создает виртуальную сеть, которая соединяет ваши контейнеры Docker на нескольких хостах или в облаках - Обеспечивает автоматическое обнаружение приложений - Сети Weave устойчивы, допускают разделение, безопасны и работают в частично связанных сетях и других неблагоприятных средах - Относительно легко настраивается

Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты

- Оптимизация

Основные инструкции в Dockerfile:
FROM — использование базового образа
RUN — выполнение команд и создание слоя образа
COPY — копирование в контейнер файлов и папок
ADD — аналогично COPY, но с распаковкой архивов
LABEL — описание метаданных
ENV — задание переменных среды
ARG — задание переменных для передачи во время сборки
CMD — описание команд с аргументами для выполнения при запуске контейнера
ENTRYPOINT — представление команды с аргументами для вызова во время выполнения
контейнера
WORKDIR — задание рабочей директории для следующей инструкции
EXPOSE — открытие порта в контейнере
VOLUME — создание точки монтирования для работы с постоянным хранилищем

Docker, оркестрация и методы их защиты

- Оптимизация

Практически каждая команда в Dockerfile создает новый слой
Больше слоев - дольше сборка - медленнее запускается и работает контейнер

```
1 FROM ubuntu
2
3 RUN apt update
4 RUN apt install python3 -y
5 RUN apt install python3-dev -y
6
```

Docker, оркестрация и методы их защиты
- Оптимизация

- сокращайте инструкци

```
1 FROM ubuntu
2
3 RUN apt update && \
4     apt install python3 python3-dev curl -y
5
```

Docker, оркестрация и методы их защиты

- Оптимизация

- очищайте кэш, в т.ч. используемых пакетных менеджеров

```
1 FROM ubuntu
2
3 RUN apt update && \
4     apt install python3 python3-dev curl -y && \
5     rm -rf /var/lib/apt/lists/*
6
```

- Изменяемые инструкции старайтесь описывать ниже остальных

(При сборке образа, инструкции считываются сверху-вниз, при наличии измененной
инструкции, оставшаяся часть инструкций будет пересобрана)

Docker, оркестрация и методы их защиты

- Оптимизация

- сокращайте количество инструкций COPY (так же, как и инструкции RUN)

```
1 FROM ubuntu
2
3 RUN apt update && \
4     apt install python3 python3-dev curl -y && \
5     rm -rf /var/lib/apt/lists/*
6
7 COPY file-1 file-2 file-3 ./dir
8
```

Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты
- Векторы атак на контейнеры

- **Уязвимый код**

```
RUN adduser ...
USER web_usr
```

• запуск от имени суперпользователя
• запуск с флагом —privileged

- **Настройка контейнеров**

Причины выполнения от имени суперпользователя внутри контейнера
• использование в контейнере распространенного ПО, требующего root-доступа (привилегированных портов)
• установка ПО
• изменение ядра

Docker, оркестрация и методы их защиты

- Векторы атак на контейнеры

- **Система сборки**

• необходимое - включить, ненужное - выкинуть
• базовый образ из белого списка, с доверенного реестра
• многоэтажная сборка
• избегайте исполняемых файлов с установленным SUID
• использование инструкции USER
• проверка монтируемых папок
• исключите чувствительные данные из образа

Docker, оркестрация и методы их защиты
- Векторы атак на контейнеры

- **Система сборки** Монтирование каталогов

```
docker run -it -v /:/hostroot ubuntu bash
```

- /etc
- /bin
- /usr/bin
- папки с логами

- сокет docker

Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты
- Векторы атак на контейнеры

- **Система сборки**

Передача чувствительных данных в контейнер

docker-compose.yml

```yaml
version: '3'

service:
  db:
    image: postgres:latest
    container_name: postgres
    restart: always
    volumes:
      - ./data:/var/lib/postgresql/data
    environment:
        POSTGRES_USER: "${DB_USER_ID}"
        POSTGRES_PASSWORD: "${DB_USER_PASSWORD}"
    ports:
      - "5432:5432"
```

Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты
- Векторы атак на контейнеры

Памятка по безопасности Docker

- OWASP Docker Top-10

• D01 - Secure User Mapping
• D02 - Patch Management Strategy
• D03 - Network Segmentation and Firewalling
• D04 - Secure Defaults and Hardening
• D05 - Maintain Security Contexts
• D06 - Protect Secrets
• D07 - Resource Protection
• D08 - Container Image Integrity and Origin
• D09 - Follow Immutable Paradigm
• D10 - Logging

• D01 — Безопасное сопоставление пользователей
• D02 – Стратегия управления исправлениями
• D03 — Сегментация сети и межсетевой экран
• D04 – Безопасные настройки по умолчанию и усиление защиты
• D05 – Поддержание контекстов безопасности
• D06 – Защитите секреты
• D07 – Защита ресурсов
• D08 — Целостность и происхождение образа контейнера
• D09 – Следуйте неизменной парадигме
• D10 — Ведение журнала

Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты

- Векторы атак на контейнеры

Best practices

• хост, контейнер, оркестратор - с последними патчами безопасности
• установлен пользователь, без флага — privileged
• доступ к сокету docker запрещен
• соблюден принцип наименьших привилегий
• используется флаг –no-new-privilege
• взаимодействие между контейнерами запрещено по умолчанию (—icc=false)
• корректное монтирование файлов и папок
• Dockerfile написан согласно требованиям безопасности
• установлен соответствующий уровень логирования

Защита инфраструктуры приложений

Docker, оркестрация и методы их защиты

- Векторы атак на контейнеры

Список материалов для изучения

1. https://cheatsheetseries.owasp.org/cheatsheets/Docker_Security_Cheat_Sheet.html
2. https://github.com/OWASP/Docker-Security/blob/main/001%20-%20Threats.md
3. https://docs.docker.com/develop/develop-images/dockerfile_best-practices/
4. https://docs.docker.com/engine/security/seccomp/
5. https://docs.docker.com/config/containers/resource_constraints/
6. https://docs.docker.com/engine/security/
7. https://docs.docker.com/engine/security/userns-remap/
8. https://docs.docker.com/engine/reference/builder/#user
9. https://access.redhat.com/blogs/766093/posts/1976473

Teach
Me
Skills

Спасибо за внимание!