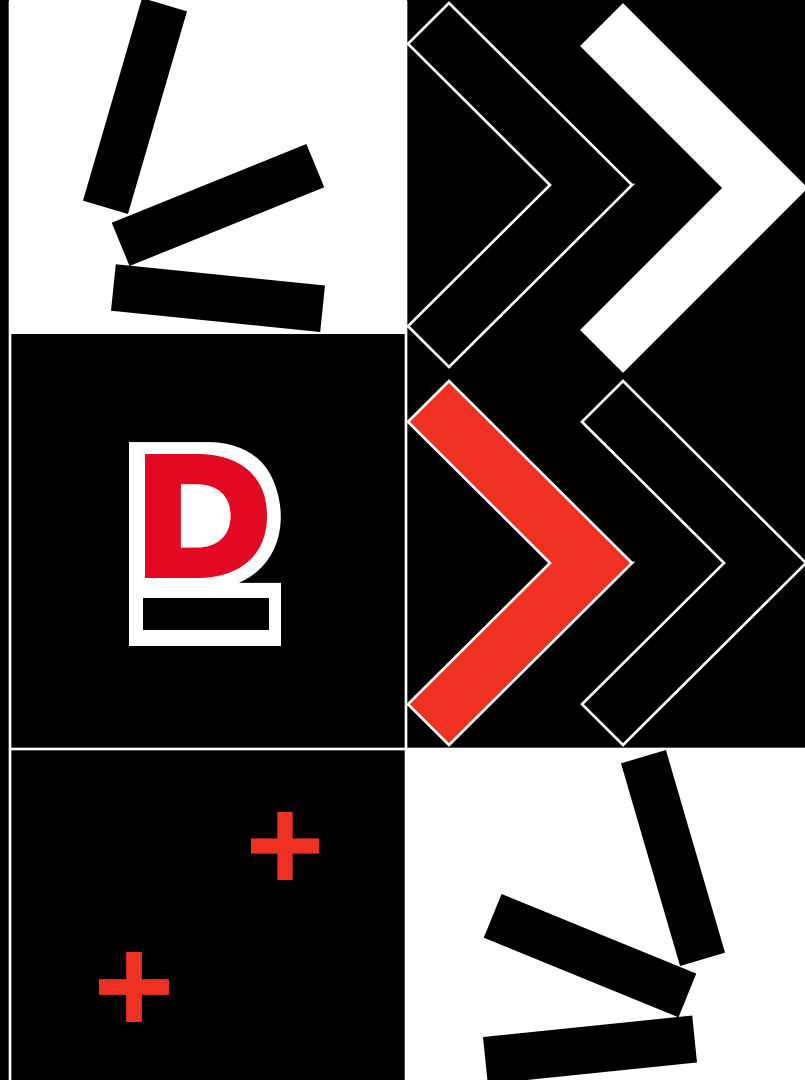
Контейнеры без права на побег: лучшие практики защиты рантайма и изоляции

Нуров Арнур
Ведущий Разработчик
Альфа-Банк



Нуров Арнур

Ведущий Разработчик

AppSec Champion

Собираю кастомные клавиатуры

Болею за Ливерпуль

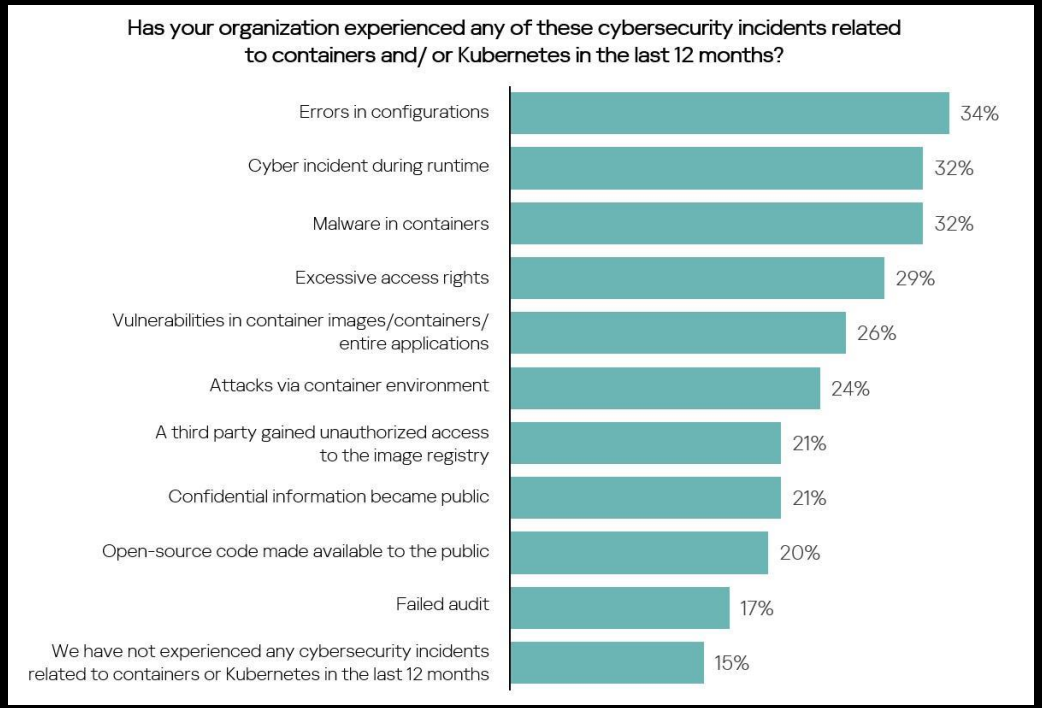
О чём доклад

Почему защита рантайма критична в 2025 году?

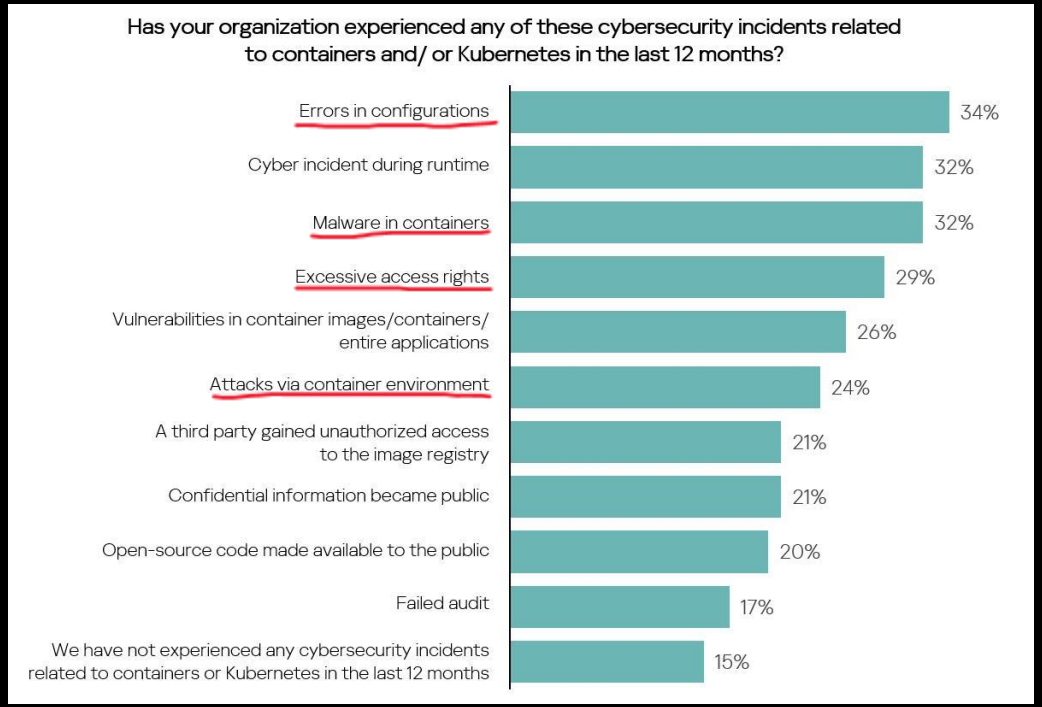
Политики безопасности Kubernetes: от PSS до Kyverno

Изоляция контейнеров: gVisor и Kata Containers

Почему защита рантайма критична в 2025 году?



Почему защита рантайма критична в 2025 году?




604

Политики безопасности Kubernetes: от PSS до Kyverno

Achtung Achtung!

Сценарий: DevOps случайно запушил в staging контейнер, в котором securityContext не был настроен — по умолчанию контейнер запустился с root-пользователем и с правом на privilege escalation.

Риск?

Если в контейнере найдётся уязвимость, это позволит вырваться из контейнера и получить доступ к хосту.

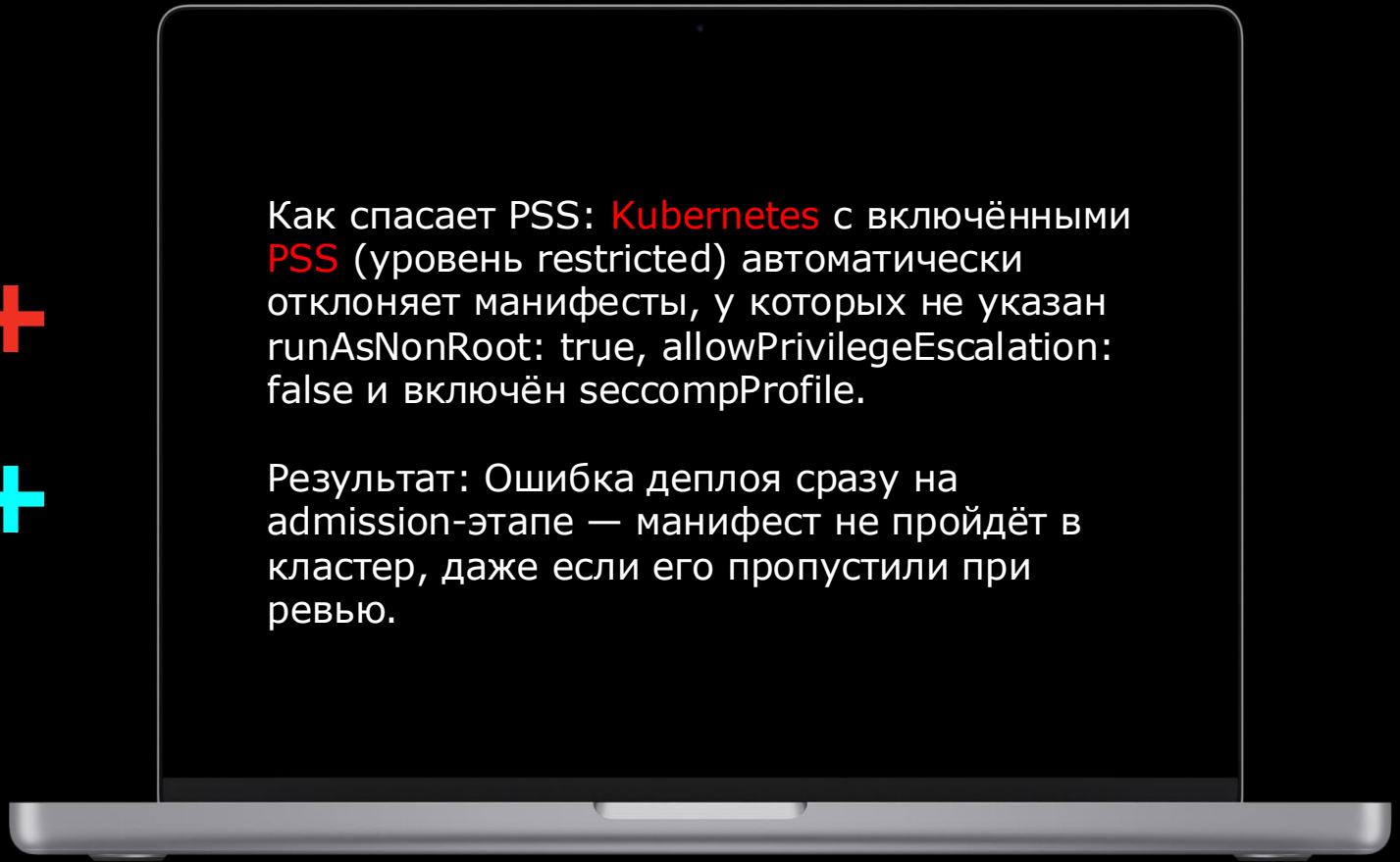
Pod Security Standarts

Feature	Privileged	Baseline	Restricted
Privileged containers	✅	❌	❌
Host networking (hostNet)	✅	❌	❌
Host path volumes	✅	❌	❌
runAsNonRoot	❌	⚠️	✅
readOnlyRootFilesystem	❌	⚠️	✅
Dangerous capabilities	✅	❌	❌
Seccomp profile required	❌	❌	✅

🎯 Когда PSS достаточно?

- Если тебе нужно быстрое, встроенное решение для базовой защиты
- Если ты не хочешь ставить дополнительные CRD и управляющий webhook
- Если ограничения restricted тебе подходят как есть

Решение и Результат



Как спасает PSS: Kubernetes с включёнными PSS (уровень restricted) автоматически отклоняет манифесты, у которых не указан runAsNonRoot: true, allowPrivilegeEscalation: false и включён seccompProfile.

Результат: Ошибка деплоя сразу на admission-этапе — манифест не пройдёт в кластер, даже если его пропустили при ревью.

Achtung Achtung!

Сценарий: Разработчик в погоне за скоростью использует образ python:3.11-slim из Docker Hub с тегом latest. Он не заметил, что этот тег недавно был обновлён — в нём теперь иная среда и потенциально небезопасные зависимости.

Риск?

Риск: Угроза поставки вредоносного образа через компрометированный Docker Hub. Также может быть несовместимость, которая сломает прод.

Kyverno

Когда PSS недостаточно гибок

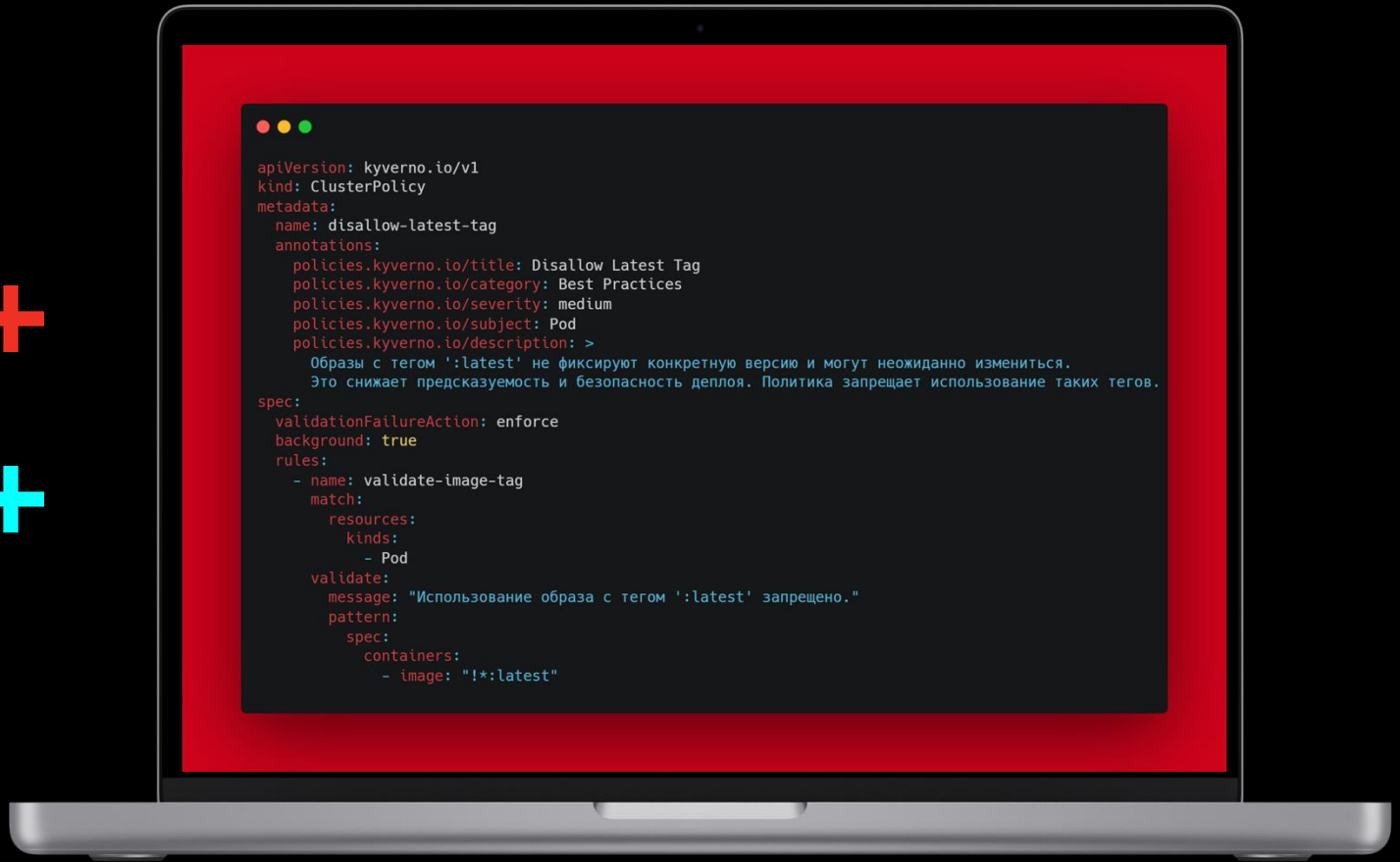
Когда нужна автоматическая починка (mutate)

Когда ты хочешь централизованный контроль

Когда надо работать не только с Pod’ами

Характеристика	Pod Security Standards (PSA)	Kyverno
Поддерживается нативно Kubernetes	✅ Да (с v1.25 включен по умолчанию)	❌ Нет, требуется установка CRDs
Уровни безопасности	Только privileged, baseline, restricted	Любые, кастомные
Гибкость и кастомизация	❌ Жестко зафиксированные уровни	✅ Абсолютно гибкие правила
Проверка дополнительных полей	❌ Только ограниченный набор securityContext	✅ Можно проверять любые поля
Возможность автозамены/патчинга	❌ Только отказ, логирование	✅ Может **автоматически исправлять** манифест
Работа с ConfigMap, Secret и др.	❌ Только Pod-специфичные ресурсы	✅ Любой ресурс Kubernetes
Поддержка шаблонов, переменных	❌	✅ Можно использовать variables, foreach и др.
Поддержка подстановки из context	❌	✅ Да, вплоть до проверки namespace и JWT
Learning/Validation mode	✅ enforce, audit, warn	✅ enforce, audit, warn

Решение и Результат



```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: disallow-latest-tag
  annotations:
    policies.kyverno.io/title: Disallow Latest Tag
    policies.kyverno.io/category: Best Practices
    policies.kyverno.io/severity: medium
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >
      Образы с тегом ':latest' не фиксируют конкретную версию и могут неожиданно измениться.
      Это снижает предсказуемость и безопасность деплоя. Политика запрещает использование таких тегов.
spec:
  validationFailureAction: enforce
  background: true
  rules:
    - name: validate-image-tag
      match:
        resources:
          kinds:
            - Pod
      validate:
        message: "Использование образа с тегом ':latest' запрещено."
        pattern:
          spec:
            containers:
              - image: "!*:latest"
```

Изоляция контейнеров: gVisor и Kata Containers

Achtung Achtung!

Сценарий: Компания предоставляет платформу, где пользователи могут запускать Python/JS скрипты в браузере (аналог Replit, JupyterHub). Скрипты запускаются в контейнерах Kubernetes.

Риск?

Пользовательский код может попытаться выполнить вредоносные команды (mount, ptrace, sh) и получить доступ к файловой системе хоста или другим подам.

gVisor

- gVisor — это легковесная, пользовательская реализация ядра Linux, которая работает как прослойка между контейнером и хост-ядром.
- Она перехватывает системные вызовы приложений и обрабатывает их в sandbox’е, не допуская прямого взаимодействия с ядром хоста.
- Проект разрабатывается и поддерживается Google.

🎯 gVisor

✖

Sentry — реализация ядра Linux в пространстве пользователя. Именно он обрабатывает системные вызовы контейнера.

✖

Gofer — компонент, который взаимодействует с файловой системой хоста и передаёт запросы от Sentry.

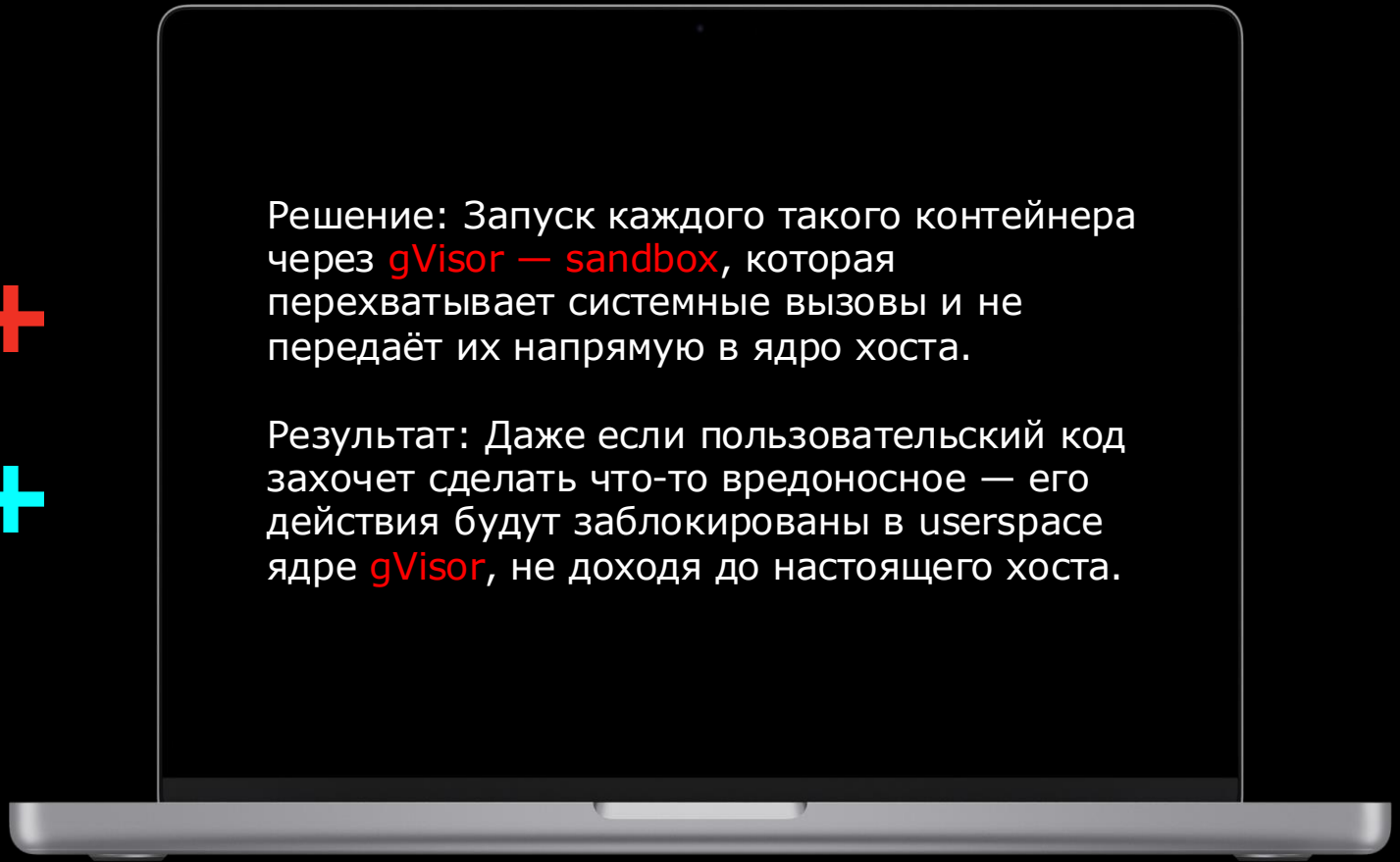
✖

ptrace mode — медленнее, но работает везде (использует ptrace для перехвата syscall’ов).

✖

KVM mode — быстрее, использует KVM для запуска sandbox’а с полной изоляцией.

Решение и Результат



Решение: Запуск каждого такого контейнера через gVisor — sandbox, которая перехватывает системные вызовы и не передаёт их напрямую в ядро хоста.

Результат: Даже если пользовательский код захочет сделать что-то вредоносное — его действия будут заблокированы в userspace ядре gVisor, не доходя до настоящего хоста.

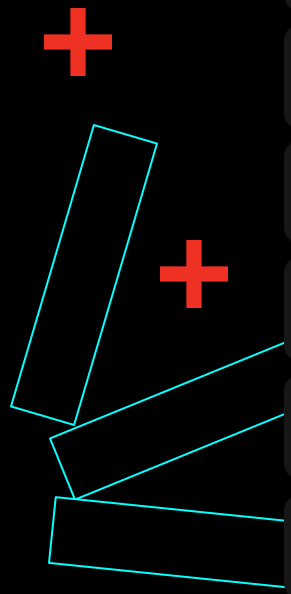
Achtung Achtung!

Сценарий: Сервис обрабатывает медицинские изображения и ML-инференс внутри микросервисов. Бэкенд работает в Kubernetes и хранит ПЕРСОНАЛЬНЫЕ ДАННЫЕ пациентов.

Риск?

Пользовательский код может попытаться выполнить вредоносные команды (mount, ptrace, sh) и получить доступ к файловой системе хоста или другим подам.

Что такое Kata Containers?



Kata Containers — это проект, который объединяет контейнерную простоту с изоляцией виртуальных машин.

Идея в том, чтобы запускать каждый контейнер внутри лёгкой виртуальной машины (VM).

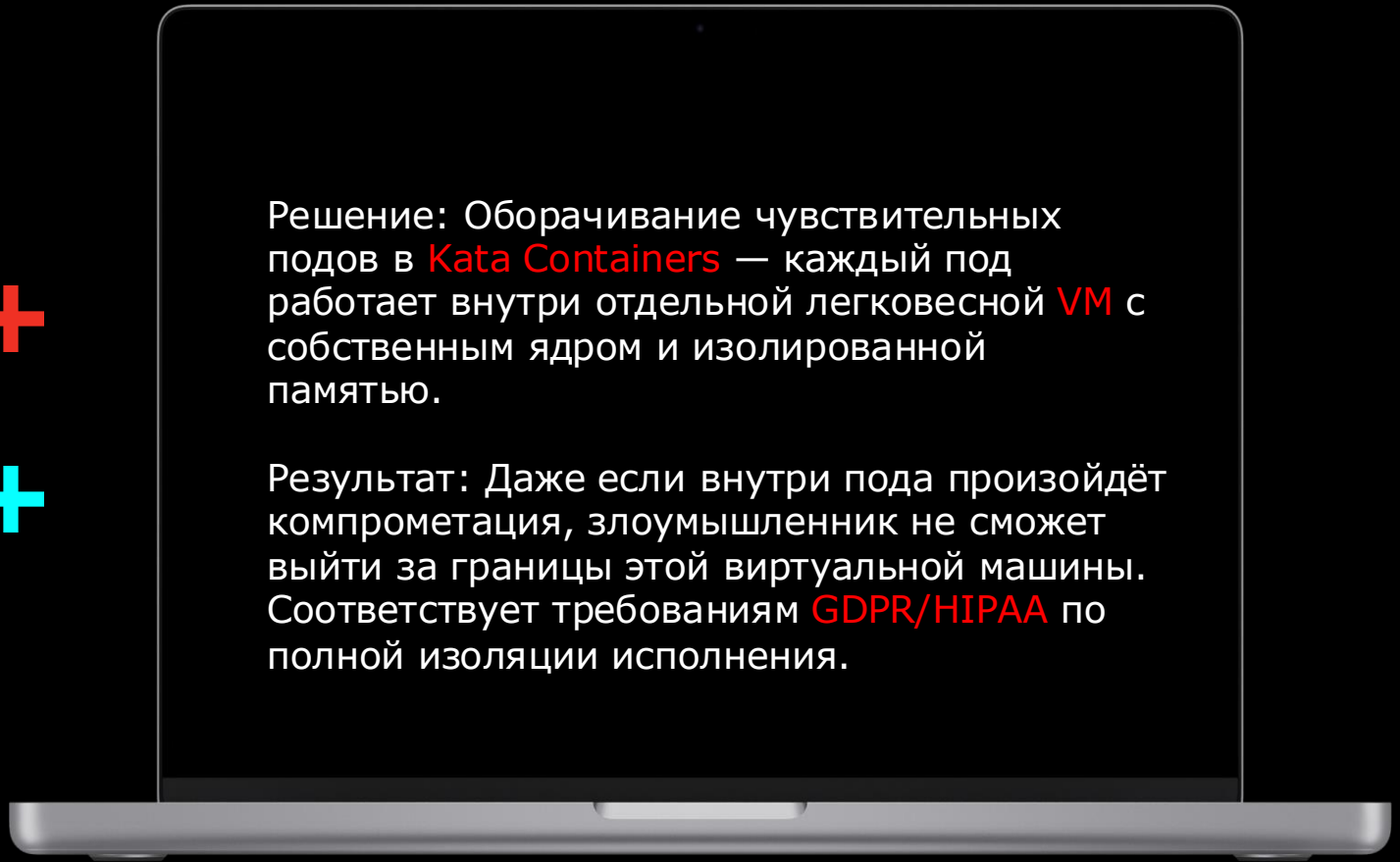
Таким образом, каждый под (или контейнер) получает: Свое Ядро

Таким образом, каждый под (или контейнер) получает: Свое Пространство Памяти

Таким образом, каждый под (или контейнер) получает: жесткую изоляцию уровня гипервизора

Проект развивается под эгидой OpenInfra Foundation (ранее OpenStack Foundation).

Решение и Результат



Решение: Оборачивание чувствительных подов в Kata Containers — каждый под работает внутри отдельной легковесной VM с собственным ядром и изолированной памятью.

Результат: Даже если внутри пода произойдёт компрометация, злоумышленник не сможет выйти за границы этой виртуальной машины. Соответствует требованиям GDPR/HIPAA по полной изоляции исполнения.

👈👉 Сравнение Kata Containers vs gVisor

Характеристика	**gVisor**	**Kata Containers**
Тип изоляции	User-space kernel (Sentry)	Hypervisor (KVM/QEMU/Firecracker)
Изоляция от хоста	Высокая	Очень высокая
Syscall handling	Эмуляция в пространстве пользователя	Полноценное ядро Linux
Поддержка POSIX API	Частичная	Почти полная
GPU поддержка	Ограничена	Лучшая, чем у gVisor
Производительность	Лучше, чем у VM, хуже чем runc	Чуть хуже gVisor, но лучше VM
Startup time	Быстрый	Дольше
Совместимость с образами	Иногда нужно адаптировать	Отличная
Потребление ресурсов	Низкое	Среднее/высокое

СПАСИБО!

Арнур Нуров

Ведущий Разработчик

@zamimaru

Alfa Digital

digital.alfabank.ru

Рассказываем о работе в IT и Digital в Альфа-Банке, делимся интересными вакансиями, новостями и полезными советами, иногда шутим