7 июня 2023 Москва, МЦК ЗИЛ

БЕКОН'23

Первая в России конференция
по БЕзопасности КОНтейнеров и контейнерных сред

AppArmor и Kubernetes: настройка проактивной защиты для безопасности приложений

Сергей Канибор

Luntry

whoami

БЕКОН

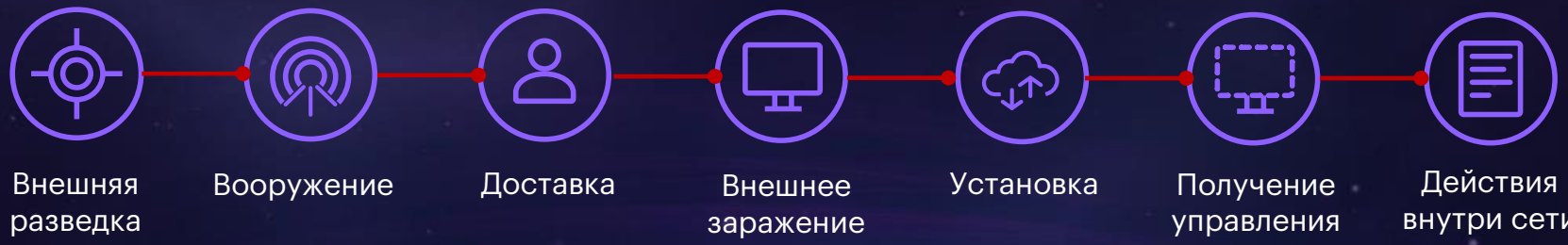
- R&D / Container Security в Luntry
- Специализируюсь на безопасности контейнеров и Kubernetes
- Спикер PHDays, VolgaCTF, HackConf, CyberCamp
- Редактор телеграм канала @k8security

Agenda

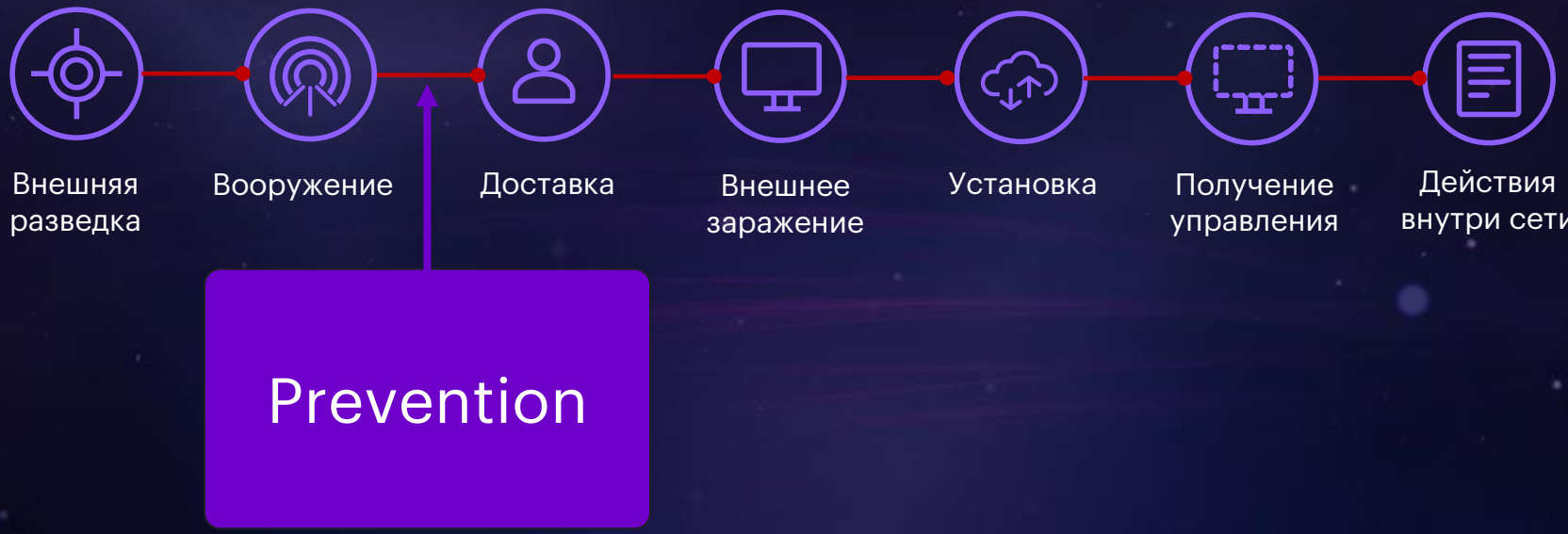
БЕКОН

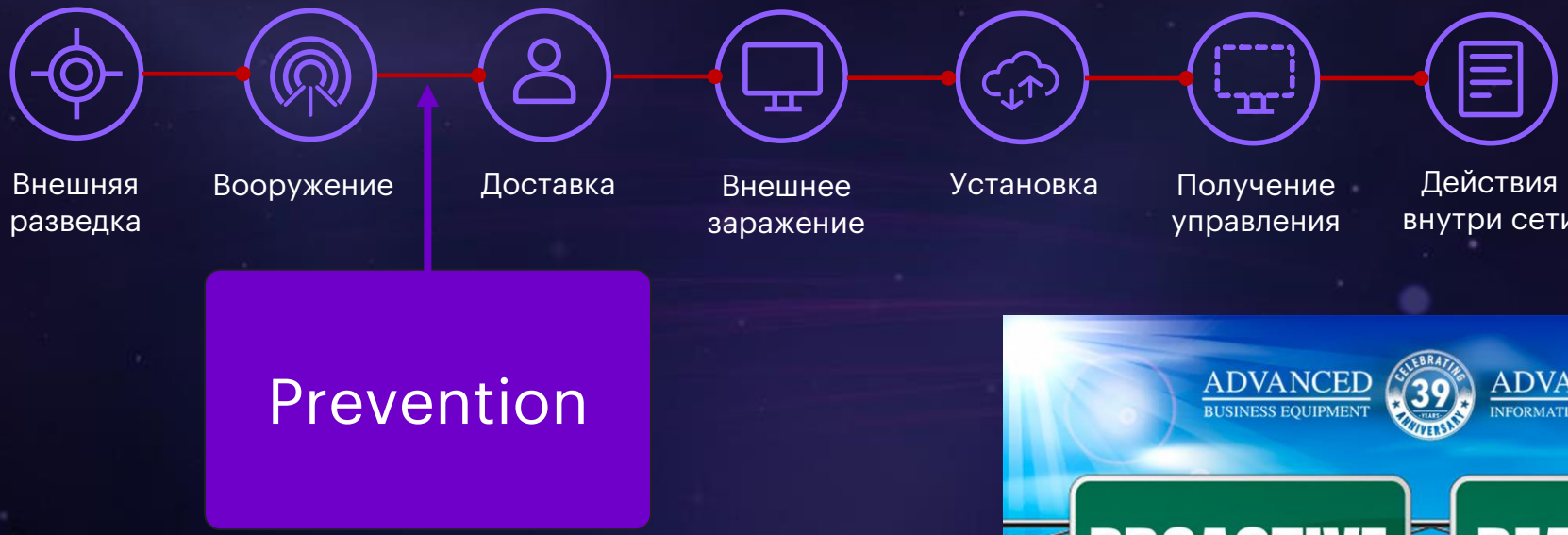
- LSM
- AppArmor – в Кубере и за его пределами
- Готовим профили
- Доставка профилей
- Дебаг и логи
- Tips & Tricks

БЕКОН


Внешняя разведка
Вооружение
Доставка
Внешнее заражение
Установка
Получение управления
Действия внутри сети

БЕКОН


Внешняя разведка
Вооружение
Доставка
Внешнее заражение
Установка
Получение управления
Действия внутри сети
Prevention


Внешняя разведка
Вооружение
Доставка
Внешнее заражение
Установка
Получение управления
Действия внутри сети
Prevention

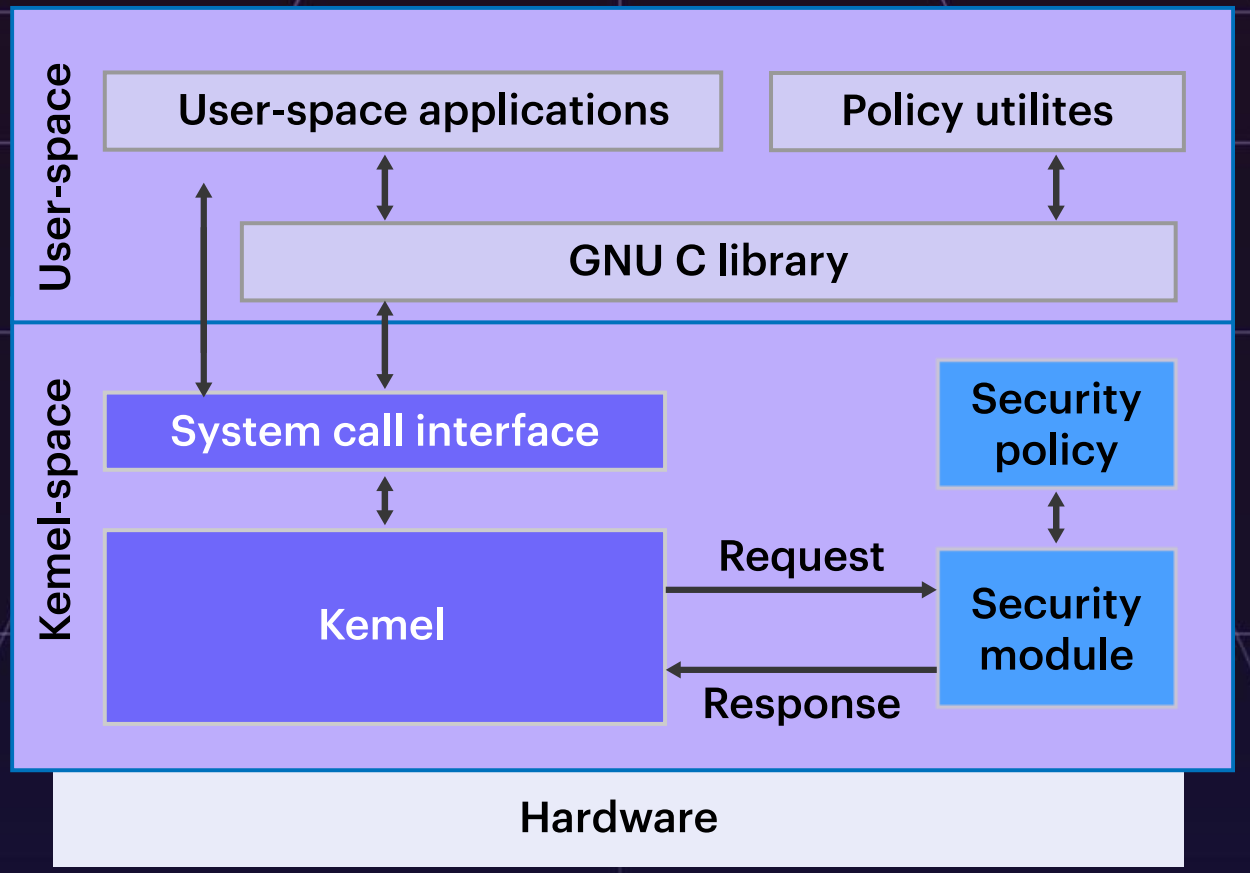

ADVANCED BUSINESS EQUIPMENT
CELEBRATING 39 YEARS ANNIVERSARY
ADVANCED INFORMATION SYSTEMS
PROACTIVE
Before Threat Detection
REACTIVE
After Threat Detection
Locates & corrects your System's potential vulnerabilities before they can be exploited & an attack can occur
• Threat Hunting
• Staff Training
• Proactive Network Monitoring
• Proactive Endpoint Monitoring
• Ethical Hacking
Bulks up your defenses against common attacks and defends against attacks that have already happened
• Firewalls
• Spam Filters
• Ad Blockers
• Password Protections
• Antivirus or Anti-Malware Software

БЕКОН

LSM

LSM – Linux Security Modules

БЕКОН



LSM – Linux Security Modules

БЕКОН

Признак сравнения	AppArmor	SELinux
Управление доступом	Path based	Files labels based
Поддержка ОС	Ubuntu, SUSE (но не все)	RHEL/Fedora (но не все)
Сложность изучения	Довольно легко	Тяжело, менее интуитивно
Возможности ограничения	Ограничивает происходящее внутри контейнера	Ограничивает то, как контейнер общается с Nodes

БЕКОН

AppArmor

AppArmor

БЕКОН

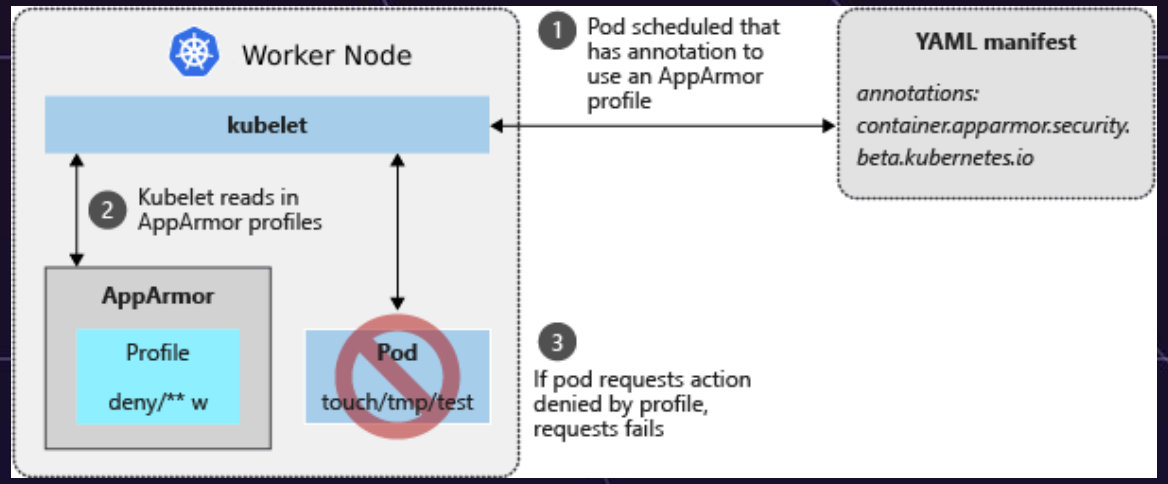
- Ограничение доступа к ресурсам:
  - Linux capabilities
  - Network access
  - File permissions
- Есть deny, allow и audit rule
- AppArmor профиль – это набор правил
- Сам профиль загружается в ядро
- Работает в двух режимах – Enforce и Complain

AppArmor в Kubernetes

БЕКОН

- Поддерживается с версии Kubernetes >= 1.4
- Требуется поддержка от самой Host OS и container runtime
- AppArmor должен быть установлен и запущен на всех Nodes
- Профиль применяется к контейнеру с помощью annotations

```
cat /sys/module/apparmor/parameters/enabled
Y
```




```
container.apparmor.security.beta.kubernetes.io/<container_name>: <profile_ref>
```

Синтаксис

БЕКОН

сетевые ограничения

выставление разрешений на файлы

ограничения на запуск бинарей

назначение capability

```
#include <tunables/global>


profile docker-nginx flags=(attach_disconnected,mediate_deleted) {
  #include <abstractions/base>

  network inet tcp,
  network inet udp,
  network inet icmp,
  deny network raw,
  deny network packet,
  file,
  umount,
  deny /bin/** wl,

  audit /** w,
  /var/run/nginx.pid w,
  /usr/sbin/nginx ix,
  deny /bin/dash mrwklx,
  deny /bin/sh mrwklx,
  deny /usr/bin/top mrwklx,


  capability chown,
  capability dac_override,
  capability setuid,
  capability setgid,
  capability net_bind_service,

}
```

БЕКОН

Готовим профили

Как получить рабочий профиль?

БЕКОН

1. Сгенерировать через дефолтные тулзы – aa-genprof
2. Использовать сторонние инструменты:
   a. в контексте K8S – KubeArmor, security-profiles operator
   b. не в контексте – bane, docker-slim
3. Обучиться на поведении приложения в контейнере => транслировать модель в AppArmor профиль

Стандартными средствами

БЕКОН

Плюсы:

- Генерация профилей из коробки
- Не нужно думать, нужно делать

Минусы:

- Генерирует профиль только “в моменте”
- Не стоит забывать, что приложения живут в контейнерах

```
Profile:  /home/packt/appackt
Execute:  /usr/bin/bash
Severity: unknown

(I)nherit / (C)hild / (N)amed / (U)nconfined / (X) ix On / (D)eny / Abo(r)t / (F)ini
sh
```

Сторонние инструменты

БЕКОН

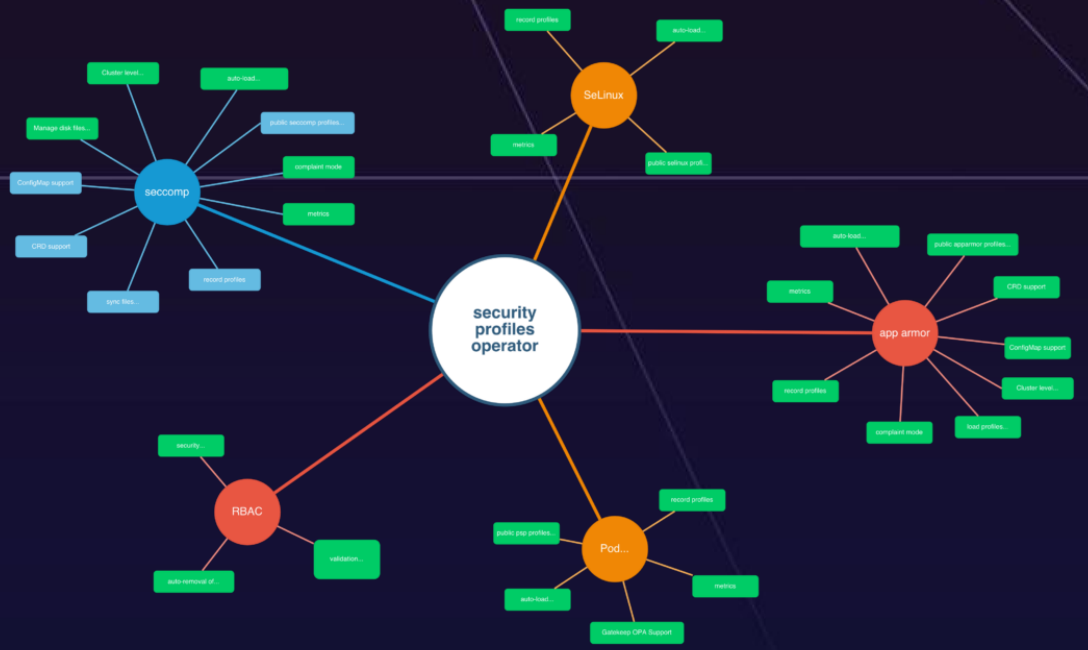
Плюсы:

- Не нужно понимать синтаксис AppArmor
- Декларативно

KubeArmor

Минусы:

- По сути облегченная версия ручного написания профилей
- Любая автоматизация зависит от покрытия кода его логики



Security Profiles Operator

БЕКОН

	Seccomp	SELinux	AppArmor
Profile CRD	Yes	Yes	Yes
ProfileBinding	Yes	No	No
Deploy profiles into nodes	Yes	Yes	Yes
Remove profiles no longer in use	Yes	Yes	Yes
Profile Auto-generation (logs)	Yes	WIP	No
Profile Auto-generation (ebpf)	Yes	No	No
Audit log enrichment	Yes	WIP	Yes

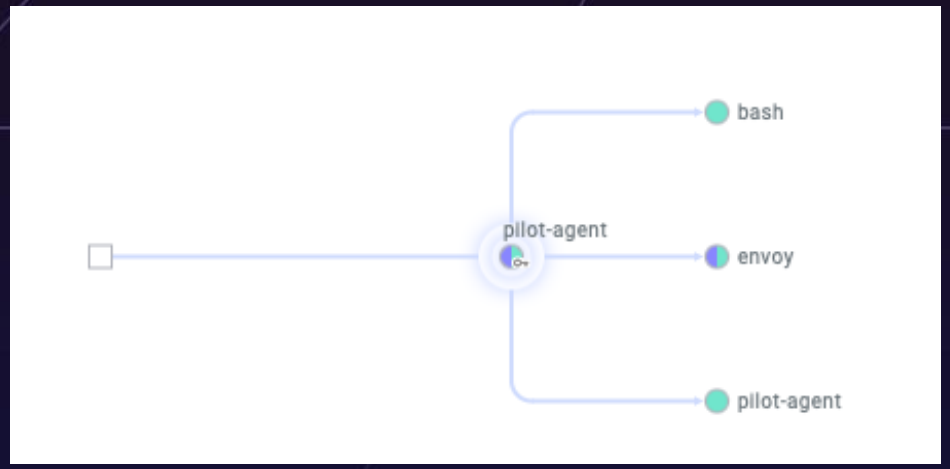
```
apiVersion: security-profiles-operator.x-k8s.io/v1alpha1
kind: AppArmorProfile
metadata:
  name: test-profile
  annotations:
    description: Block writing to any files in the disk.
spec:
  policy: |
    #include <tunables/global>

    profile test-profile flags=(attach_disconnected) {
      #include <abstractions/base>

      file,

      # Deny all file writes.
      deny /** w,
    }
```

AppArmor профиль по behaviour model

БЕКОН

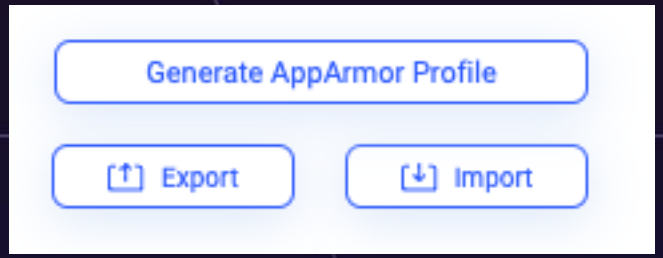
Плюсы:

- Мы почти на 100% уверены, что профиль валидный и не сломает контейнер через какое то время
- Не нужно знать специфики



Минусы:

- Нужно какое-то время на обучение модели
- Нужен инструмент



Общие нюансы

БЕКОН

- В AppArmor профиле можно задавать ограничения для разных процессов, но в контейнере обычно всё проще
- Deny правила не могут быть переопределены Allow правилами
- Нужно как-то доставлять профили на Nodes
- Само собой предварительно нужно всё отдебажить

БЕКОН

Доставляем профили

Доставка профилей

БЕКОН

- AppArmor профили должны быть раскинуты по всем Nodes, где запускаются контейнеры
- Как это сделать?
  - Руками
  - Скриптом
  - Kubernetes operator

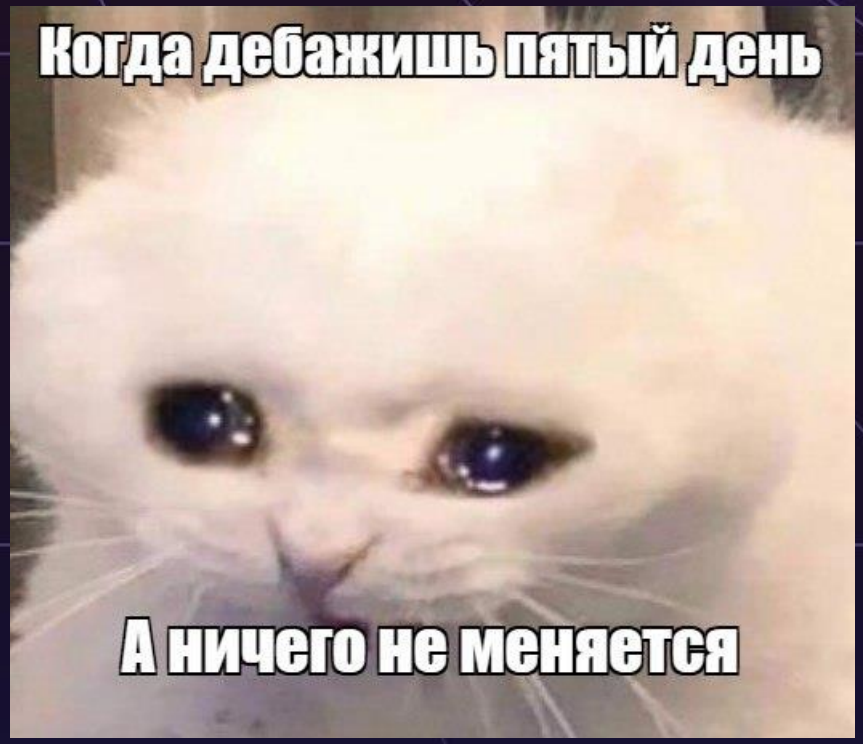
БЕКОН

Дебажим профили

Debug

БЕКОН

- Применяем профили в Complain mode
  - aa-complain /path/to/bin
- /etc/apparmor/parser.conf
  - loglevel debug
- /var/log/kern.log, /var/log/syslog, /var/log/apparmor/*
- Используем dmesg или journalctl
- Утилиты из пакета AppArmor
  - aa-status
  - aa-complain
  - aa-logprof



БЕКОН

Tips & tricks

AppArmor shebang bypass

БЕКОН

```
profile testprofile {
    file,
    capability,
    network,
    unix,
    signal,
    /** ix,
    audit deny /usr/bin/perl
rwxmlk,
}
```

Issue

```
# cat /root/script.sh
#!/usr/bin/perl
print "hi\n";

hi!
```

securityContext >> AppArmor

БЕКОН

```
securityContext:
  capabilities:
    add: ["NET_ADMIN", "SYS_ADMIN", "SYS_MODULE"]
    drop:
      - all
```

securityContext >> AppArmor

БЕКОН

```
#include <tunables/global>

profile k8s-apparmor-example-caps flags=(attach_disconnected) {
  #include <abstractions/base>

  file,

  deny capability net_admin,
  deny capability sys_admin,
  deny capability sys_module,
}
```

securityContext >> AppArmor

БЕКОН

```
root@mtkpi-pod:/run# capsh --print
WARNING: libcap needs an update (cap=40 should have a name).
Current: =
Bounding set =cap_net_admin,cap_sys_module,cap_sys_admin
```

На самом деле нет

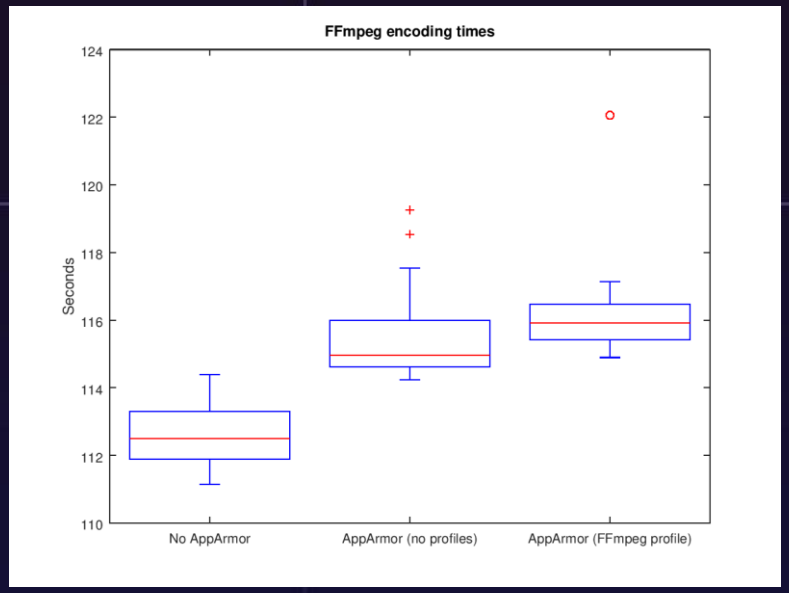
БЕКОН

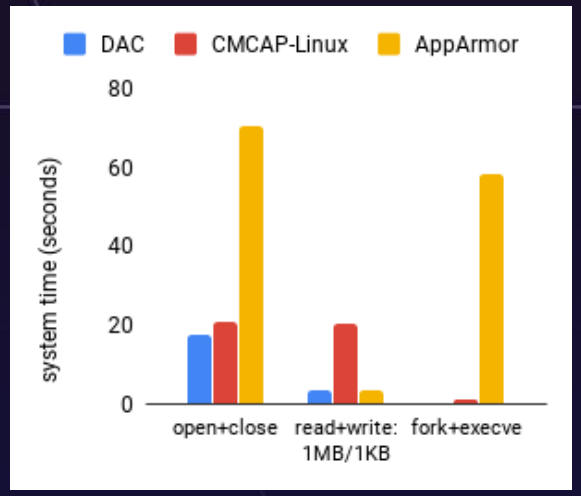
```
root@ubuntu:/# unshare -UrmC bash # create new user and cgroups namespaces
unshare: unshare failed: Operation not permitted
root@ubuntu:/#
```

Что насчет нагрузки?

БЕКОН

- Безусловно она есть
- Всё зависит от того, что делает ваш код внутри контейнера
- AppArmor работает через LSM и поэтому трекается каждый syscall
- Нужно породить очень много syscalls, чтобы почувствовать на себе





[AppArmor and Its Performance Impact]

[Does AppArmor decrease the system performance?]

Выводы

БЕКОН

- Вокруг AppArmor можно построить мощный prevention
- AppArmor не единственный способ это сделать:
  - Network Policy
  - Distroless image
  - Policy Engine
- Как и любая другая технология требует времени для изучения

Полезные ссылки

БЕКОН

- Restrict a Container's Access to Resources with AppArmor
- AppArmor and Kubernetes
- Debugging Apparmor
- Русскоязычный гайд на 2,5 часа
- Building the Largest Working Set of Apparmor Profiles

Tg: rObinak

@k8security

Site: luntry.ru