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

R&D/Container security, Luntry

whoami

БЕКОН

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

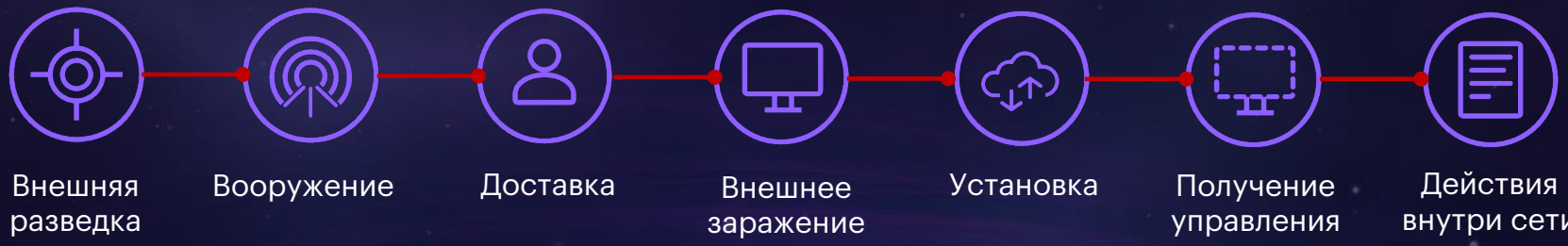
Agenda

БЕКОН

- Linux Security Modules (LSM)
- AppArmor – в Кубере и за его пределами
- Готовим AppArmor профили
- Дебаг и логи профилей
- Доставка профилей
- Tips & Tricks

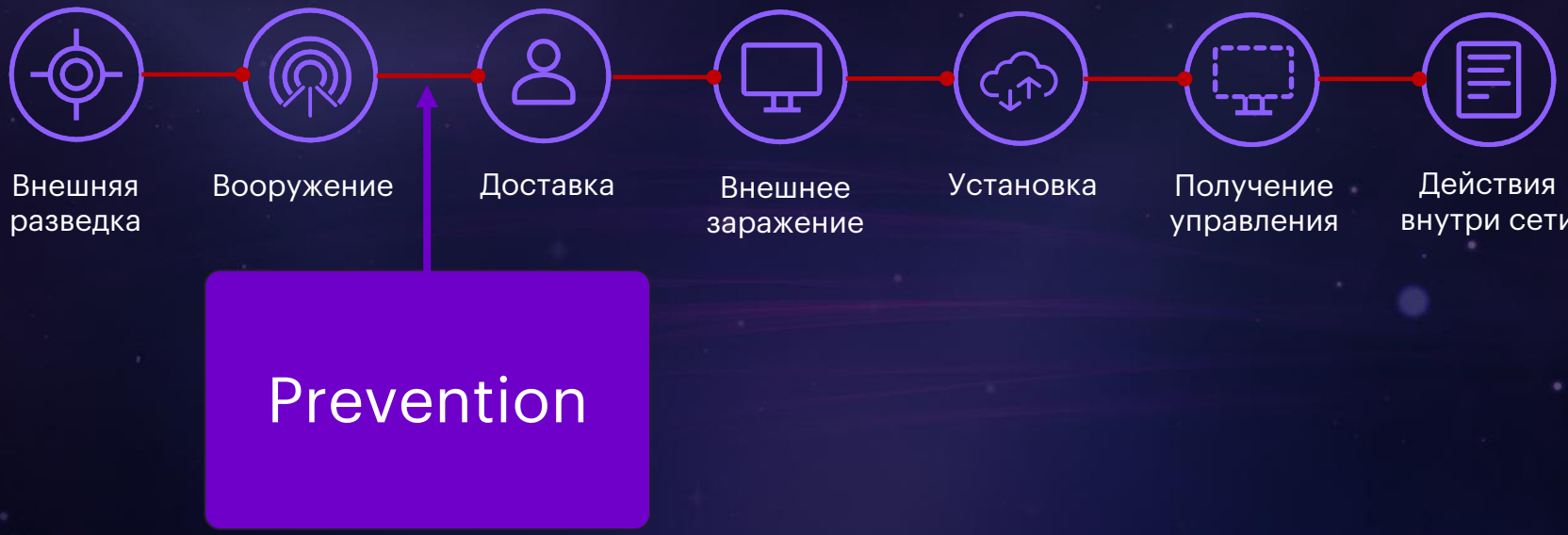
Cyber-Kill Chain

БЕКОН



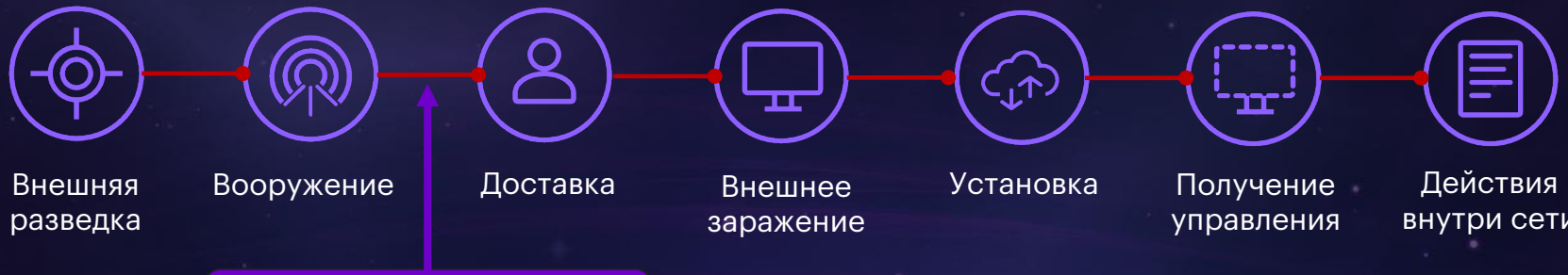
Cyber-Kill Chain

БЕКОН



Cyber-Kill Chain

БЕКОН

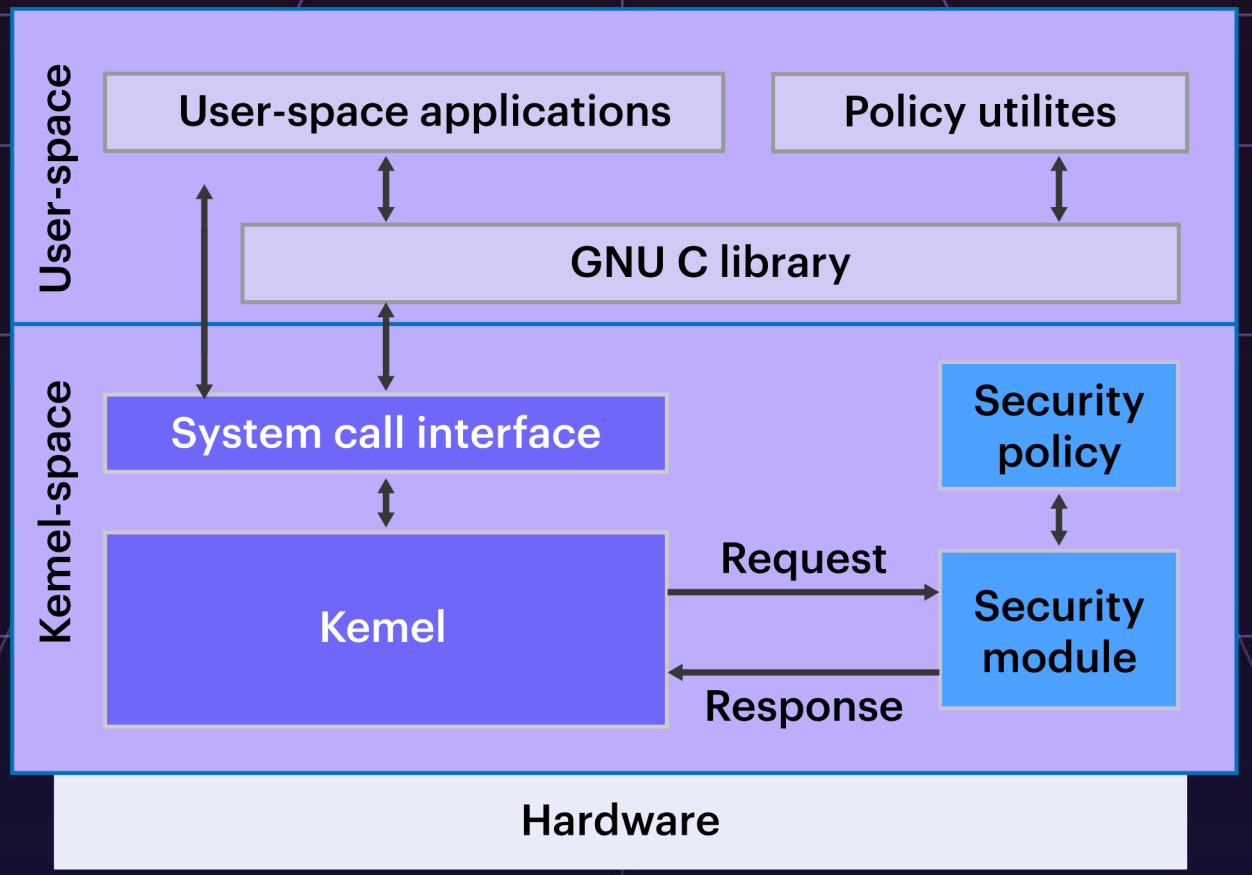


БЕКОН

LSM

LSM – Linux Security Modules

БЕКОН



LSM – Linux Security Modules

БЕКОН

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

БЕКОН

AppArmor

AppArmor

БЕКОН

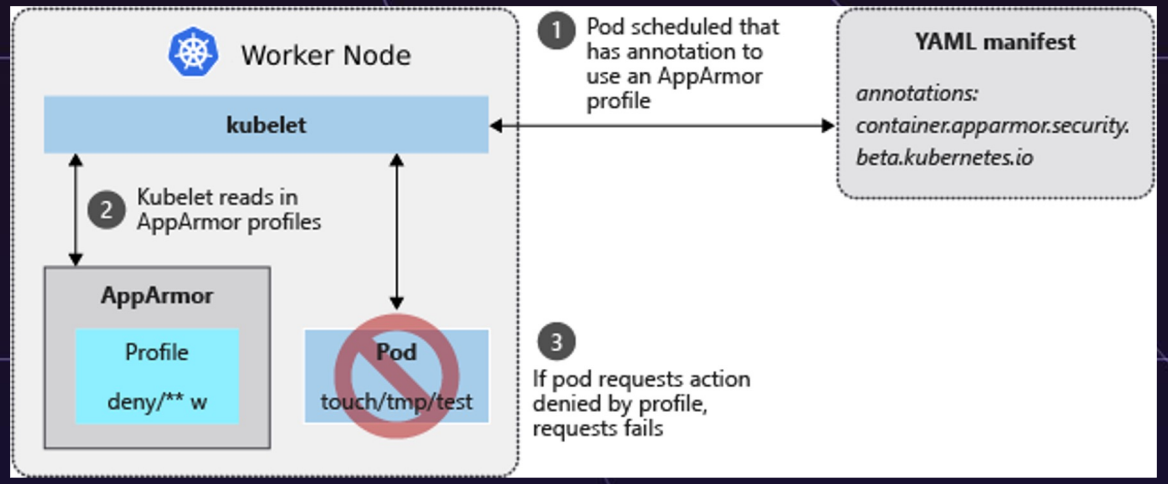
- Ограничение доступа к ресурсам:
  - Linux capabilities
  - Network access
  - File permissions
- Есть deny, allow и audit rule
- AppArmor профиль – это набор правил
- Сам профиль загружается в ядро
- Работает в двух режимах – Enforce и Complain

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>
```

Пример профиля

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,

}
```

Права на файлы

БЕКОН

- r – чтение
- w – запись
- ix – наследование выполнения
- a – запись в конец файла
- k – блокировка файла
- l – создание симлинков
- m - загрузка бинарей в память
- cx – переход в профиль нижнего уровня при выполнении
- Cx – переход в профиль нижнего уровня с очисткой переменных окружения
- px – требуется определение дискретного профиля безопасности
- Px – аналогично + очистка переменных окружения
- ux – не проверять запуск новых процессов
- Ux – аналогично + очистка переменных окружения

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 модели поведения приложения в контейнере и конвертировать модель в AppArmor профиль

Стандартными средствами

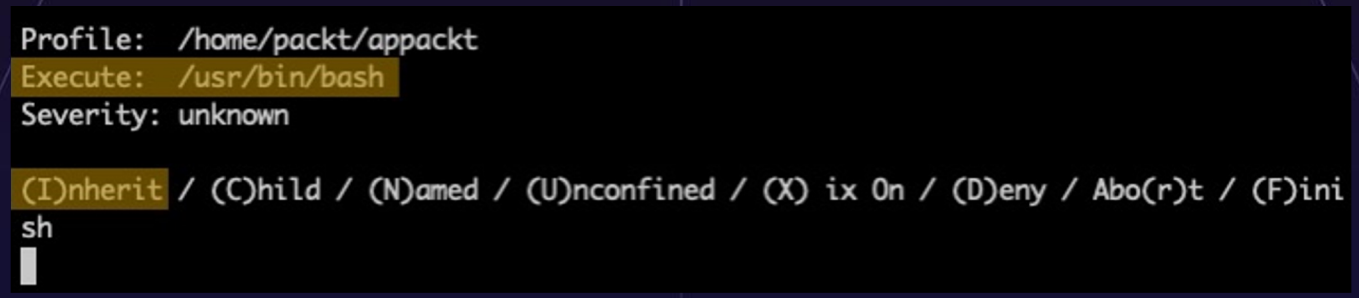
БЕКОН

Плюсы:

- Генерация профилей из коробки
- Понимает все возможные права

Минусы:

- Генерирует профиль только “в моменте”
- Не понимает сущности контейнер/микросервис
- Высокий порог входа



Сторонние инструменты

БЕКОН

Плюсы:

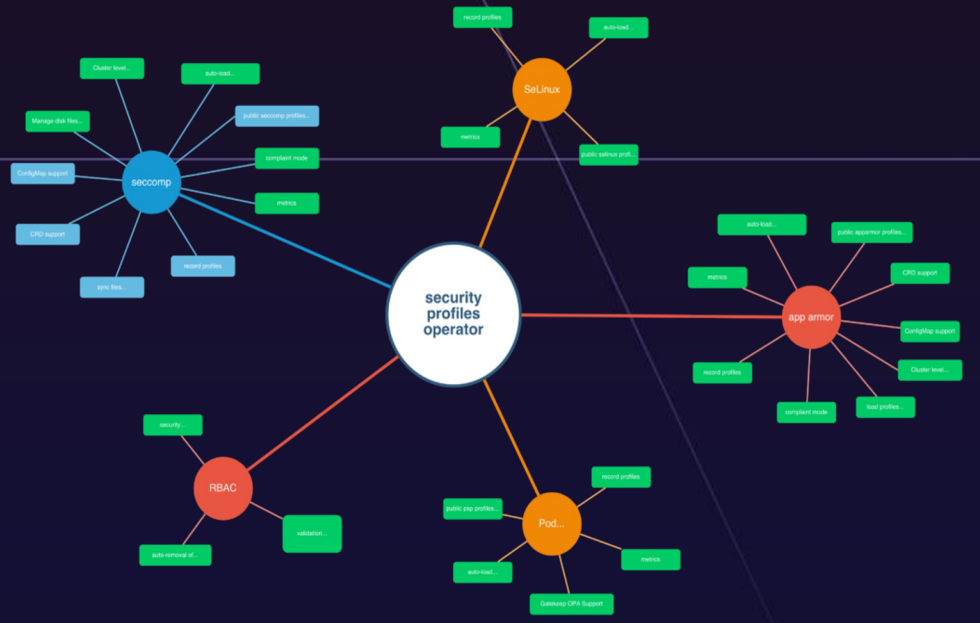
- Низкий порог входа
- Декларативно (CRD/YAML)
- Понимают сущности контейнер/микросервис

KubeArmor

*очень сырой, много багов

Минусы:

- Генерирует профиль только «в моменте»
- Использует только базовые права



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

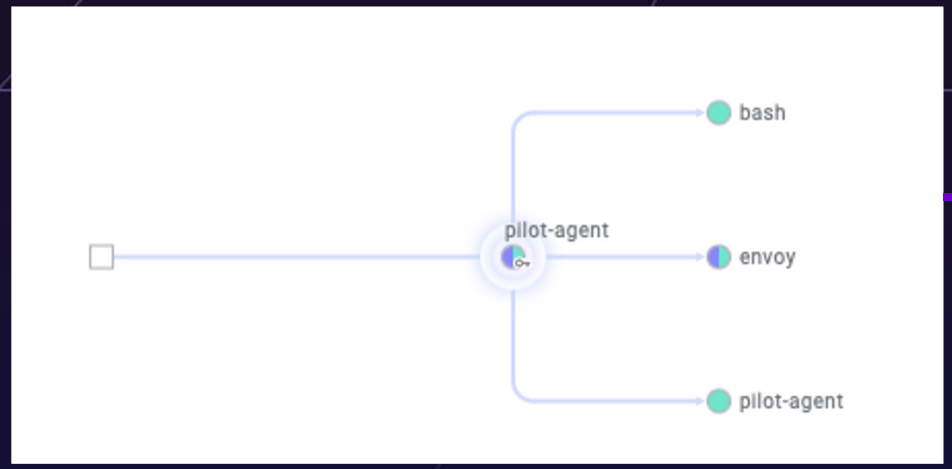
БЕКОН

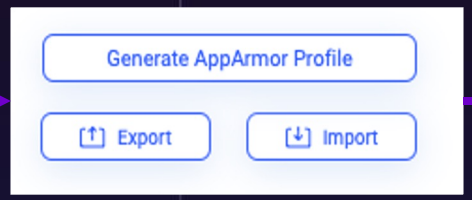
Плюсы:

- Более высокий уровень точности
- Низкий порог входа
- Понимают сущности контейнер/микросервис

Минусы:

- Время обучения зависит от поведения
- Используем только базовые права






```
#include <tunables/global>
profile k8s-apparmor-istio-proxy flags=(attach_disconnected) {
  #include <abstractions/base>
  file,
  network,
  deny /** x,
  /usr/local/bin/pilot-agent ix,
  /usr/bin/bash ix,
  /usr/local/bin/envoy ix,
}
```

Download

Общие нюансы

БЕКОН

- Не оставляйте возможности перезаписывать исполняемые файлы
- Deny правила не могут быть переопределены Allow правилами
- Само собой предварительно нужно всё отладить
- Нужно как-то доставлять профили на Nodes

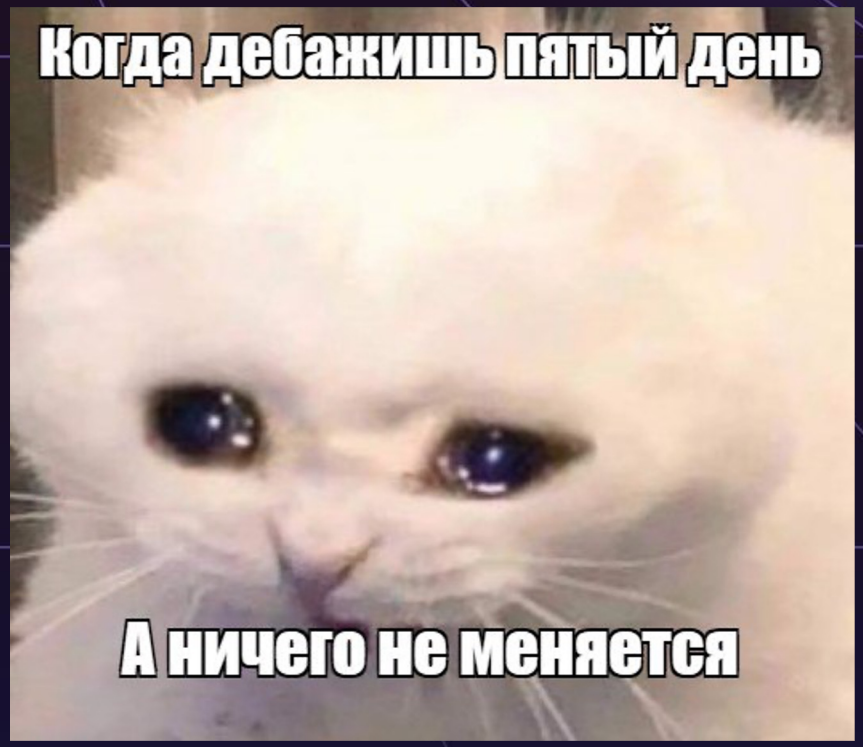
БЕКОН

Отлаживаем профили

Debug

БЕКОН

- Применяем профили в Complain mode
  - apparmor-parser –C –W /path/to/profile
- /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

Доставляем профили

Доставка профилей

БЕКОН

- AppArmor профили должны быть раскинуты по всем Nodes, где запускаются контейнеры
- Как это сделать?
  - Руками
  - Скриптом
  - Kubernetes operator
  - Ansible/Chef/Puppet/Salt/...

БЕКОН

Tips & tricks

AppArmor shebang bypass

БЕКОН

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

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

hi!
```

Issue

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

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

Выводы

БЕКОН

- Вокруг AppArmor можно построить мощный prevention
- AppArmor отлично сочетается с другими механизмами:
  - Network Policy
  - readOnly filesystem
  - 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 (документация K8S)
- AppArmor and Kubernetes (блоговая заметка про AppArmor & K8S)
- Debugging Apparmor (wiki Ubuntu)
- Русскоязычный гайд на 2,5 часа (видео гайд на русском)
- Building the Largest Working Set of Apparmor Profiles (свежий доклад от The Linux Foundation)
- AppArmor and Its Performance Impact (про нагрузку)
- Does AppArmor decrease the system performance?


7 июня 2023 Москва, МЦК ЗИЛ
Первая в России конференция
по БЕзопасности КОНтейнеров и контейнерных сред
БЕКОН
LUNTRY
Tg: r0binak
@k8security
Site: luntry.ru