MeetUp Cyber Camp

Форензика для контейнеров и контейнерных инфраструктур

Дмитрий Евдокимов

Founder&CTO, Luntry

Обо мне

> Я не верю в то, что систему можно сделать надежной и безопасной, не понимая того, как она устроена.

Основатель и технический директор Luntry

Более 15 лет опыта в ИБ

Специализация – безопасность контейнеров и Kubernetes

Автор ТГ-канала k8s (in) security

Эксперт в сфере безопасности контейнерных сред

- Организатор конференции «БеКон» по БЕзопасности КОНтейнеров
- Бывший редактор рубрик в журнале «ХАКЕР», автор серии статей
- Автор курса «Cloud Native безопасность в Kubernetes»
- Член программного комитета CFP DevOpsConf и HighLoad++

Спикер

VK Kubernetes	Confidence	ZeroNights	БеКон	HITB
DevOpsConf	HackInParis	KuberConf	BlackHat	PHDays
Kazhackstan	HighLoad++	OFFZONE	DevOops	SAS

О компании Luntry

Luntry – это комплексная защита на всем жизненном цикле контейнерных приложений и средств оркестрации на базе Kubernetes

Продукт в реестре Минцифры:

https://reestr.digital.gov.ru/reestr/1057835/

В процессе получения сертификата ФСТЭК

План доклада

1. ВВЕДЕНИЕ

2. ВЕРХНЕУРОВНЕВЫЙ ВЗГЛЯД

3. ПОГРУЖЕНИЕ НА КОНТЕЙНЕРНЫЙ УРОВЕНЬ

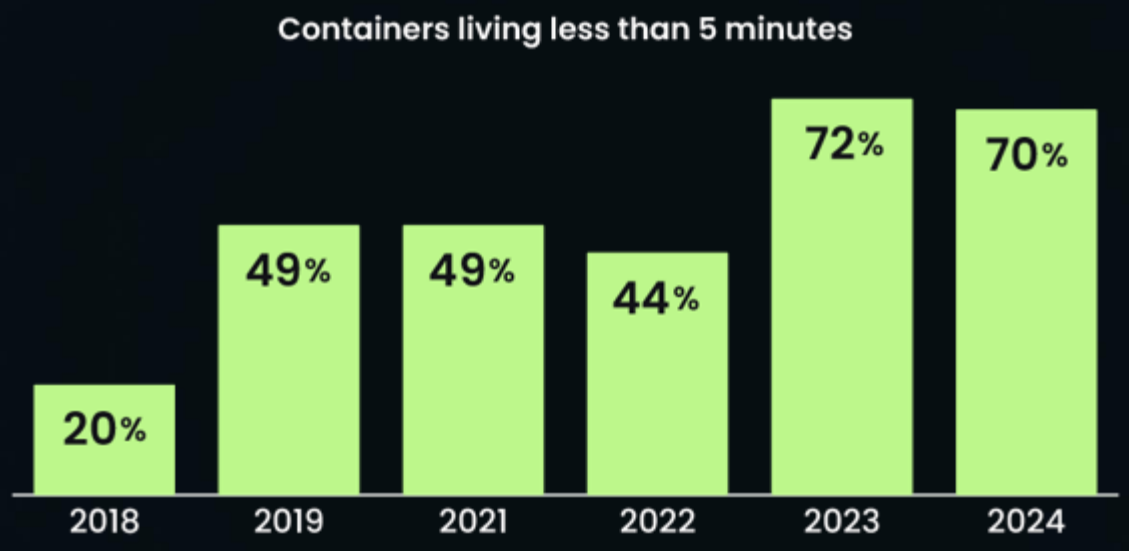
4. ЗАКЛЮЧЕНИЕ

ВВЕДЕНИЕ

**В этой презентации мы рассматриваем кейс,
когда инцидент уже произошел,
и повлиять на него невозможно.**

Помните, что предотвращение всегда дешевле расследования ;)

Динамическое окружение

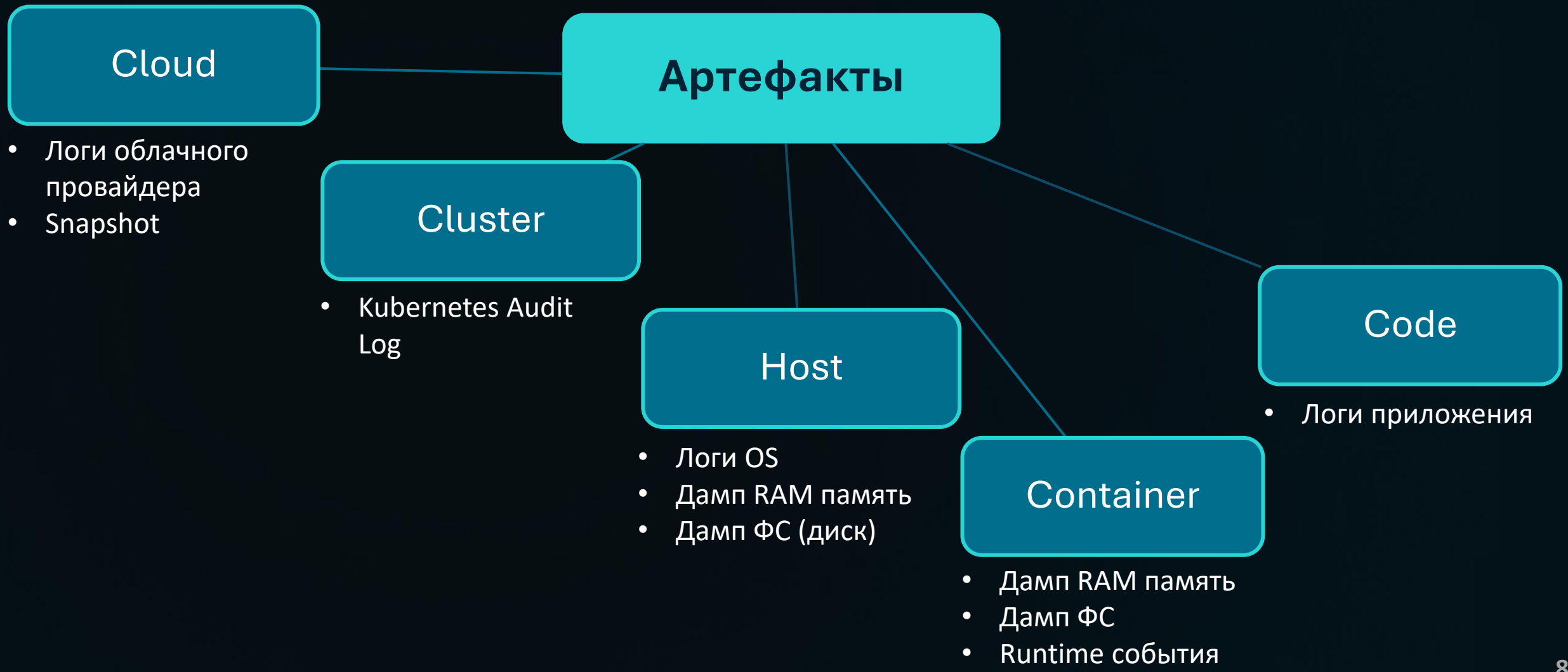


Sysdig 2024 Cloud-Native Security and Usage Report

- Малый срок жизни контейнеров
- K8s: Self-healing, self-control
- Множество уровней абстракций
- Следы злоумышленника в контейнере пропадают сами

Источники данных

(следы и артефакты)



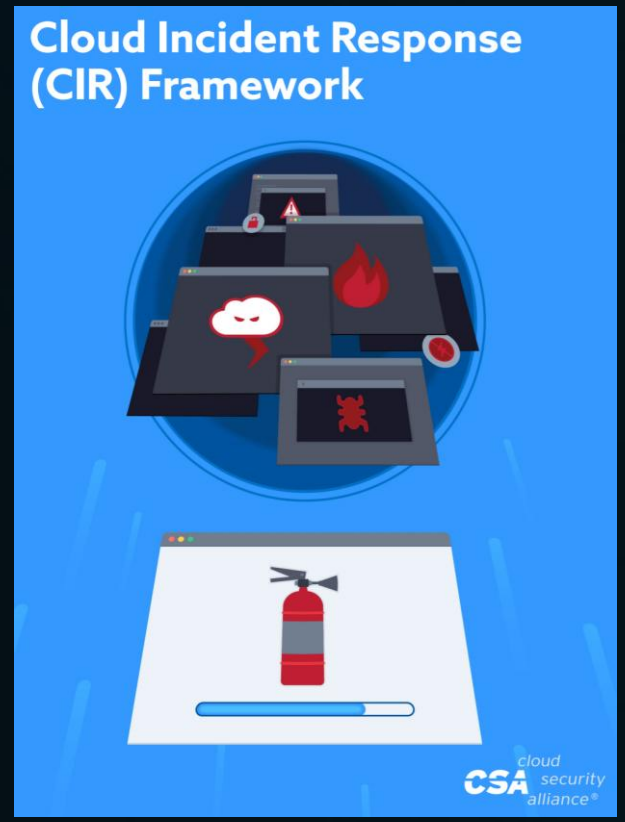
ВЕРХНЕУРОВНЕВЫЙ ВЗГЛЯД

Уровень Cloud

1. **Изучаем логи облачного провайдера**
   (так же как и любые другие логи)

2. **По возможности, делаем snapshot нужного хоста**

3. **Все зависит от реализация облачного провайдера**
   - Включено?
   - Цена?
   - Детализация?
   - Формат?
   - Срок хранения?
   - …



Cloud Incident Response Framework

Уровень Cluster

На этом уровне можно анализировать:

1. События от Policy Engine

- Нужно использовать Kyverno, OPA Gatekeeper
- Политики могут работать в audit и enforce режимах
- Изучаем, как и любые другие логи

2. Kubernetes Audit Log

- По умолчанию выключен
- Нужно заранее корректно настроить Audit Policy
- Изучаем, как и любые другие логи

“Kubernetes Audit Log в арсенале SOC”, SOC Forum 2024.

Уровень Host

1. Linux хост

- Актуально все, что актуально для всех Linux
- Он может быть автоматически перезагружен/отключен/обновлен/… со всеми артефактами и следами
- Огромное количество хороших работ по форензике Linux

Advanced Linux Detection and Forensics CheatSheet by Defensive Security

2. Учитываем специфичные директории от Kubernetes

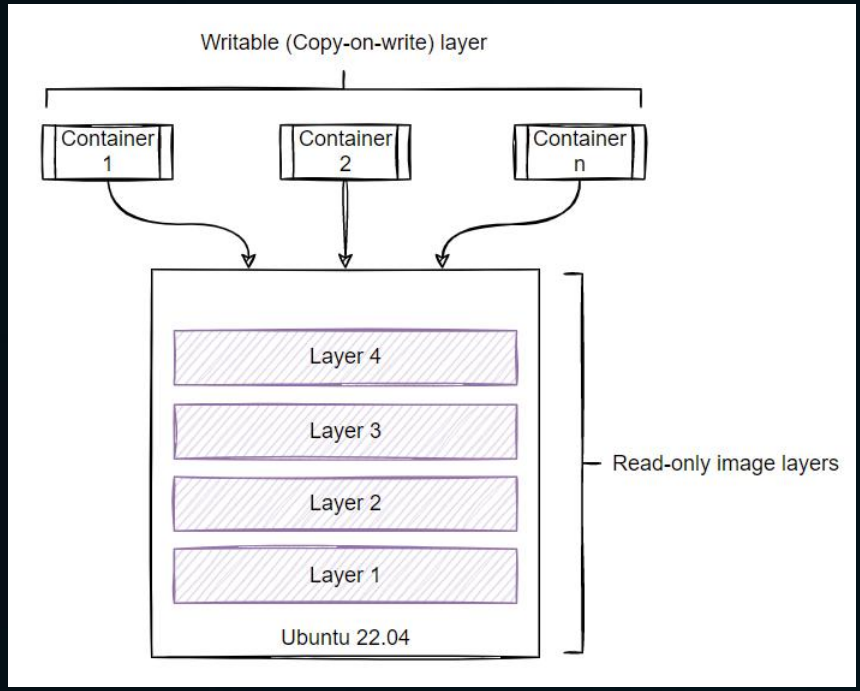
- Логи системных компонентов на Master и Worker Nodes (Kube-proxy, Kubelet, etcd и т.д.)
- Так как Kubernetes – это фреймворк, расположение этих директорий может варьироваться

Уровень Container

1. Классический Linux Container
Linux process + cgroup + namespaces (pid, user, uts, ipc, net, mnt, …) + pivot_root + image

2. Контроль Runtime событий

3. Container Image
Неизменяемый пакет файлов операционной системы, кода приложения и любых зависимостей приложения Union File System
-> OverlayFS как реализация



4. **Контроль запуска только разрешенных images**

```
root      598309  0.0  0.0 110128  6224 ?        Sl   Nov20   0:07  \_ containerd-shim -namespace moby -workdir /var/lib/containerd/io.containerd.runtime.v1
root      598334  1.4  5.5 7236340 1832196 pts/0 Ssl+ Nov20  39:39      \_ /docker-java-home/bin/java -Djava.util.logging.config.file=/opt/atlassian/conflue
root      599854  1.0  1.3 7007820 427956 pts/0  Sl+  Nov20  28:11      |   \_ /usr/lib/jvm/java-8-openjdk-amd64/jre/bin/java -classpath /opt/atlassian/conf
root      701694  0.0  0.0   4288   764 ?        Ss+  Nov20   0:00      \_ /bin/sh
```

Уровень Code

1. **Изучаем логи приложения**
   (как и любые другие логи)

2. **Изучаем на ВПО и ПО двойного назначения**
   - LotL-атака
   - GTFOBins

3. **Изучаем уязвимости**
   - Потенциальные точки проникновения и нанесения ущерба

4. **Изучаем SBOM**
   - Возможно, есть скомпрометированные компоненты
     -> Supply chain атака

ПОГРУЖЕНИЕ
НА КОНТЕЙНЕРНЫЙ УРОВЕНЬ

Без паники!

1. **Выводим Node из шедулинга**
   kubectl cordon

2. **Не заходим на Node или container**
   Чтобы не занести ничего лишнего

3. **Не завершаем, не перезапускаем**
   Чтобы не затереть артефакты и следы

4. **Изолируем Pod по сети**
   Deny all NetworkPolicy

5. **Убеждаемся, что побег и атаки на рядом стоящие сервисы не возможны**
   - Анализ SecurityContext
   - Анализ техник не через SecurityContext

Базовые утилиты для Live взаимодействия

1. Для работы с Kubernetes

Kubectl

2. Для взаимодействия с контейнерами

- Docker • ctr • crictl • nerdctl • podman • Cdebug
- Ephemeral Containers (Kubernetes >= 1.25 (stable)

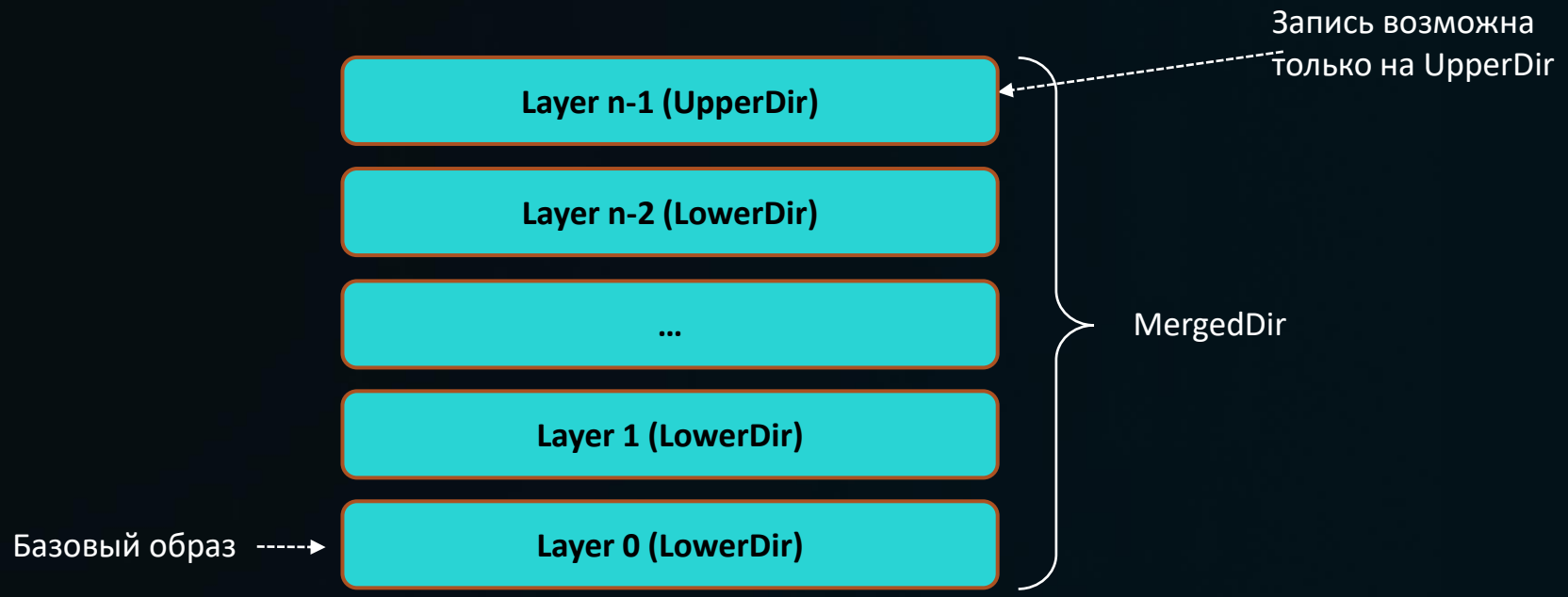
```
ubuntu@i-0b2aa6818f893983b:~$ sudo crictl logs 549604ded2540
AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using 100.96.1.143. Set the 'ServerName' directive globally to su
age
AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using 100.96.1.143. Set the 'ServerName' directive globally to su
age
[Wed Jan 25 21:39:37.338520 2023] [mpm_event:notice] [pid 1:tid 140207943607424] AH00489: Apache/2.4.49 (Unix) configured -- resuming normal operations
[Wed Jan 25 21:39:37.338684 2023] [core:notice] [pid 1:tid 140207943607424] AH00094: Command line: 'httpd -D FOREGROUND'
172.20.40.173 - - [26/Jan/2023:09:40:45 +0000] "POST / HTTP/1.1" 200 45
100.96.3.123 - - [26/Jan/2023:11:17:46 +0000] "GET / HTTP/1.1" 200 45
100.96.4.165 - - [26/Jan/2023:11:17:46 +0000] "GET / HTTP/1.1" 200 45
100.96.3.123 - - [26/Jan/2023:11:17:46 +0000] "GET / HTTP/1.1" 200 45
```

Дамп ФС контейнера

1. **Нет смысла дампать всю ФС целиком**
2. **Нижние слои могут очень много весить**
3. **Злоумышленник может взаимодействовать только с upper layer**

Обратите внимание:

- /proc/<pid>/root
- /proc/<pid>/mountinfo
- overlayfs
- UpperDir



Container Explorer

- https://github.com/google/container-explorer
- **Разработка Google**
- **Написано на Go**
- **Поддерживает container и docker**
- **Помогает проводить offline форензику ФС снимка Node**

Container Explorer provides the following functionalities:

- Exploring namespaces
- Exploring containers
- Exploring images
- Exploring snapshots
- Exploring contents
- Exploring container drift
- Mounting containers
- Support JSON output

diffoci

- https://github.com/reproducible-containers/diffoci
- Написан на Go
- Инструмент для сравнения Docker и OCI образов

```
$ diffoci diff docker://golang:1.21-alpine3.18 docker://my-golang-1.21-alpine3.18 --semantic --report-d
INFO[0000] Loading image "docker.io/library/golang:1.21-alpine3.18" from "docker"
docker.io/library/golang:1.21 alpine3.18        saved
Importing       elapsed: 2.6 s  total:   0.0 B  (0.0 B/s)
INFO[0004] Loading image "docker.io/library/my-golang-1.21-alpine3.18:latest" from "docker"
docker.io/library/my golang 1.21 alpine3        saved
Importing       elapsed: 2.6 s  total:   0.0 B  (0.0 B/s)
TYPE     NAME                      INPUT-0
Layer    ctx:/layers-1/layer       length mismatch (457 vs 454)
File     lib/apk/db/scripts.tar    eef110e559acb7aa00ea23ee7b8bddb52c4526cd394749261aa244ef9c6024a4
Layer    ctx:/layers-1/layer       name "usr/local/share/ca-certificates/.wh..wh..opq" only appears in
Layer    ctx:/layers-1/layer       name "usr/share/ca-certificates/.wh..wh..opq" only appears in input
Layer    ctx:/layers-1/layer       name "etc/ca-certificates/.wh..wh..opq" only appears in input 0
Layer    ctx:/layers-2/layer       length mismatch (13927 vs 13926)
Layer    ctx:/layers-2/layer       name "usr/local/go/.wh..wh..opq" only appears in input 0
File     lib/apk/db/scripts.tar    073bb5094fc5bba800f06661dc7f1325c5cb4250b13209fb9e3eaf4e60e4bfc4
Layer    ctx:/layers-3/layer       length mismatch (4 vs 3)
Layer    ctx:/layers-3/layer       name "go/.wh..wh..opq" only appears in input 0
```

Дамп RAM памяти

1. Linux контейнер это упрощенные Linux-процессы с определёнными свойствами

2. Дампим память процессов как и всегда:

- /proc/<pid>/maps
- /proc/<pid>/mem
- ptrace
- gdb dump memory

3. Полезные инструменты для анализа памяти:

- AVML
- Volatility
- Rekall

Forensic Container Checkpointing в k8s

1. **Базируется на Checkpoint/Restore In Userspace (CRIU)**
2. **Требуется включение ContainerCheckpoint feature gate на API server**
3. **С Kubernetes v1.25 (alpha), v1.30 (beta) включено по умолчанию**
4. **Требуется поддержка на стороне Container Runtime и установленный CRIU**
   - CRI-O v1.25.0+ (с включенным параметром enable_criu_support)
   - containerd v2.0+
   - *Podman 0.10.1+
5. **Для создания checkpoint требуется обращение к kubelet/checkpoint эндпоинту**
6. **Полученный Checkpoint можно восстановить как в Kubernetes, так и за его пределами**

```
curl -X POST "https://localhost:10250/checkpoint/namespace/podId/container"
```

checkpointctl

- https://github.com/checkpoint-restore/checkpointctl
- Написан на Go
- Для работы с checkpoints контейнеров

```
# Start vulnerable web application
$ sudo podman run --name dsvw -p 1234:8000 -d quay.io/rst0git/dsvw

# Perform arbitrary code execution attack: $(echo secret)
$ curl "http://localhost:1234/?domain=www.google.com%3B%20echo%20secret"
nslookup: can't resolve '(null)': Name does not resolve

Name:      www.google.com
Address 1: 142.250.187.228 lhr25s34-in-f4.1e100.net
Address 2: 2a00:1450:4009:820::2004 lhr25s34-in-x04.1e100.net
secret

# Create a checkpoint for forensic analysis and leave the container running
$ sudo podman container checkpoint --leave-running -l -e /tmp/dsvw.tar

# Analyse checkpoint memory to identify the attacker's injected code
$ sudo checkpointctl memparse --pid 1 /tmp/dsvw.tar | grep 'echo secret'
00007faac5711f60  6f 6d 3b 20 65 63 68 6f 20 73 65 63 72 65 74 00  |om; echo secret.|
```

```
$ checkpointctl inspect /tmp/ubuntu_looper.tar.gz --ps-tree --metadata

awesome_booth
├── Image: docker.io/library/ubuntu:latest
├── ID: 695b77deb38281244a114da111e2ee606ab9464ffa94a98be382d181c2121c9c
├── Runtime: crun
├── Created: 2023-03-08T08:45:33+03:00
├── Engine: Podman
├── Checkpoint size: 2.8 MiB
├── Root FS diff size: 309.0 KiB
├── Metadata
│   └── Annotations
│       ├── io.container.manager: libpod
│       └── org.opencontainers.image.stopSignal: 15
└── Process tree
    └── [1]  bash
        └── [5]  su
            └── [6]  bash
                └── [47]  loop.sh
                    └── [74]  sleep
```

Runtime события

1. **Нужно видеть, понимать и детектировать происходящее в контейнере.**

Иначе никаких следов/артефактов для расследования может вообще не быть

2. Runtime контроль

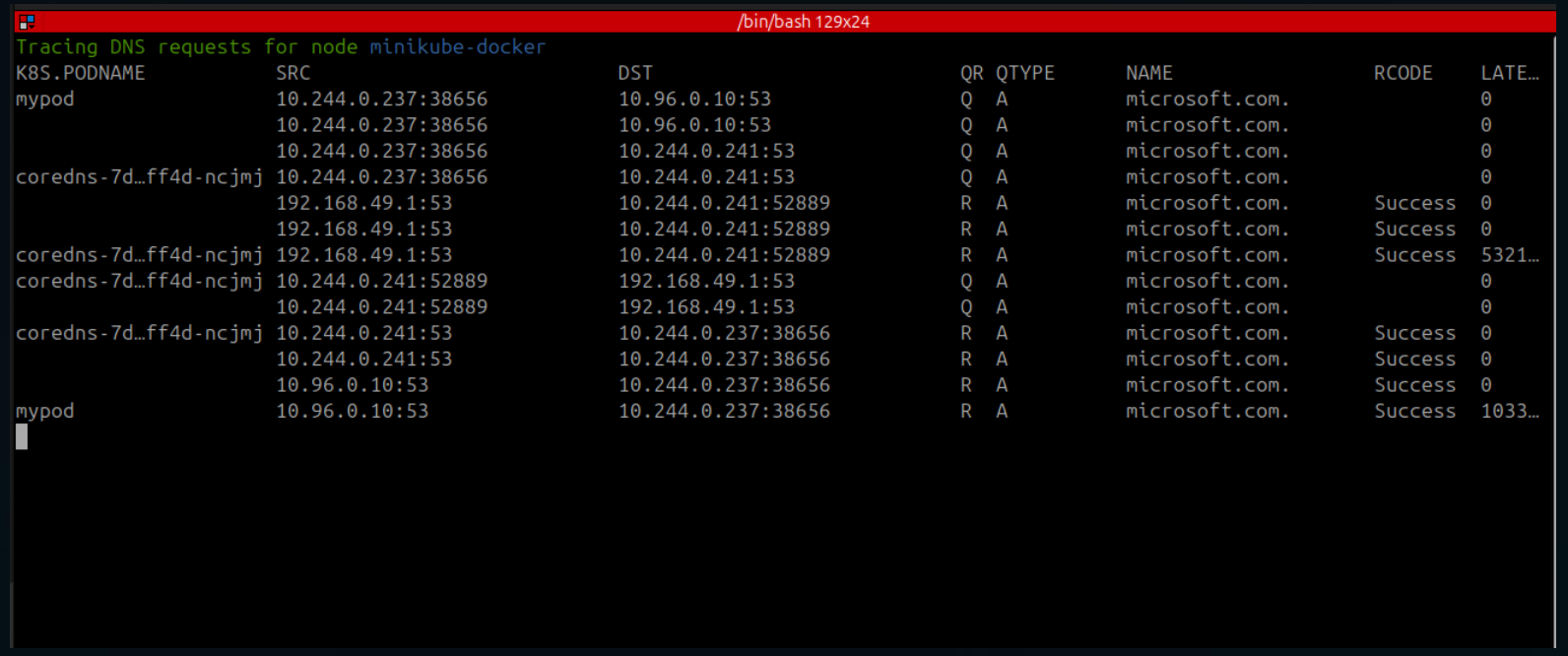
- На основе профиля поведения
- На основе правил

3. Контроль запуска исполняемых файлов отсутствующих в исходном образе

- Executables drift
- Запуск с Upper layer

Inspektor Gadget

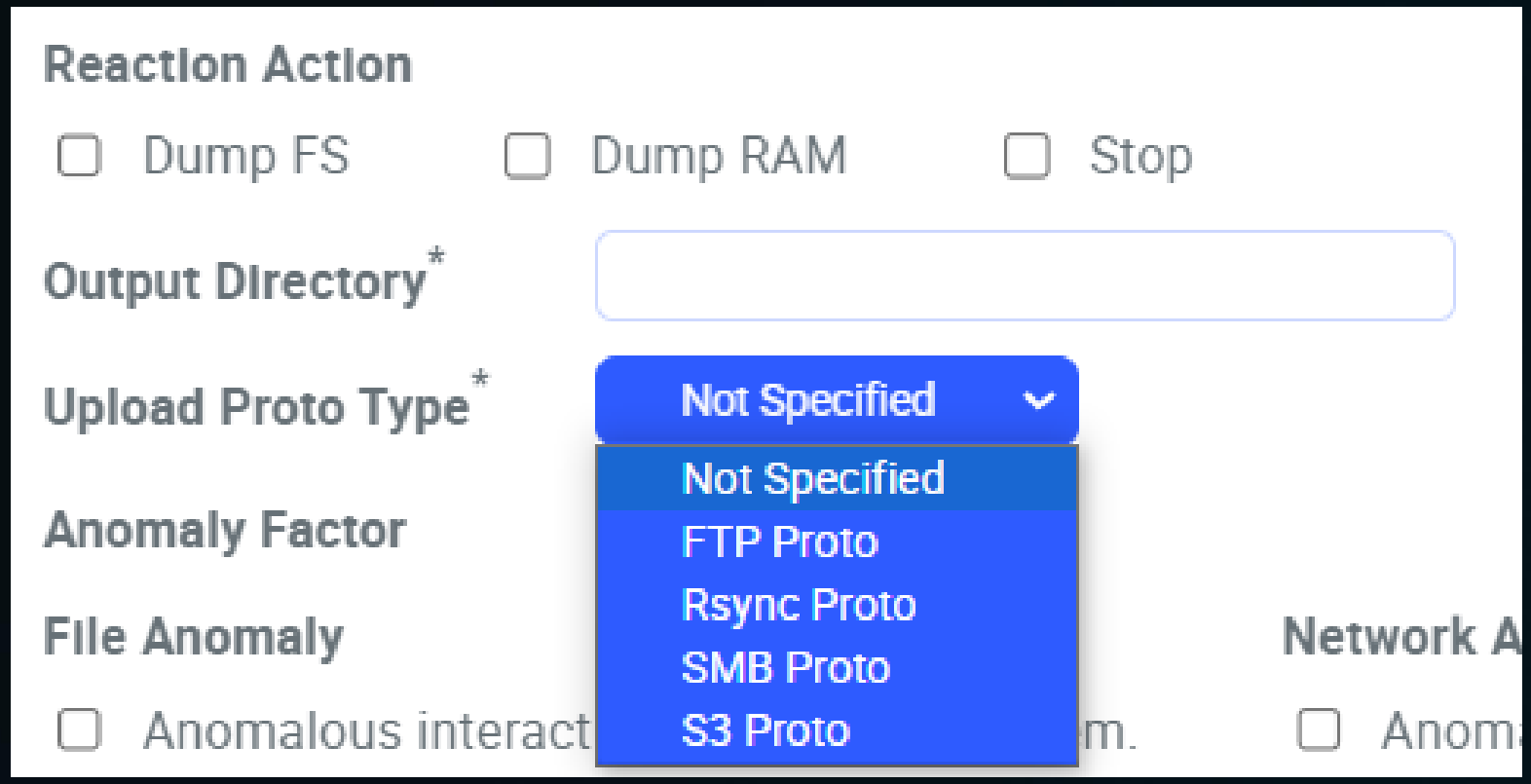
- https://github.com/inspektor-gadget/inspektor-gadget
- Написан на Go с использование eBPF
- Полезен для live форензики
- Представляет из себя набор утилит для сбора информации



Отправка артефактов

Выносим все собранные артефакты за пределы кластера для дальнейшего анализа

- FTP
- Rsync
- SMB
- S3
- …



Реагирование?!

1. Атакующий мог породить другие потоки, оставить для себя бэкдоры
2. Такой контейнер — это скомпрометированная среда

ЗАКЛЮЧЕНИЕ

Выводы

1. Необходимо фиксировать инциденты в контейнерах максимально быстро, пока они еще существуют
2. Сочетайте классические приемы форензики с характерными для контейнеров
3. Для эффективной форензики в контейнерных средах нужно использовать специфику контейнеров
4. Важно собирать артефакты для расследования в контейнерных окружениях

Полезные материалы

1. Ловим злоумышленников и собираем улики в контейнерах Kubernetes
   Вебинар Luntry 2024
2. Поймай меня, если сможешь: Как обнаружить следы злоумышленника в Kubernetes инфраструктуре
   BI.ZONE Cybersecurity Meetup 2024
3. Специфика расследования инцидентов в контейнерах
   KazHackStan 2022: Toitarys
4. State of Checkpoint/Restore in Kubernetes
   FOSDEM 2025
5. CSI Forensics: Unraveling Kubernetes Crime Scenes
   Cloud Native SecurityCon North America 2024
6. CSI Container: Can You DFIR It?
   Cloud Native SecurityCon North America 2023
7. Container Forensics: What to Do When Your Cluster is a Cluster
   KubeCon + CloudNativeCon 2019
8. Challenges in Cloud Native Forensics
   KubeCon + CloudNativeCon North America 2021
9. Forensic container analysis
   Kubernetes blog 2023
10. Forensic container checkpointing in Kubernetes
    Kubernetes blog 2022

Спасибо за внимание!

Дмитрий Евдокимов

Founder&CTO Luntry

Email: de@luntry.ru

Twitter: @evdokimovds

@Qu3b3c

Channel: @k8security

Site: www.luntry.ru

luntry.ru

k8security

luntrysolution