Поймай меня, если сможешь: Как обнаружить следы злоумышленника в Kubernetes инфраструктуре

Сергей Канибор

R&D / Container Security, Luntry

whoami

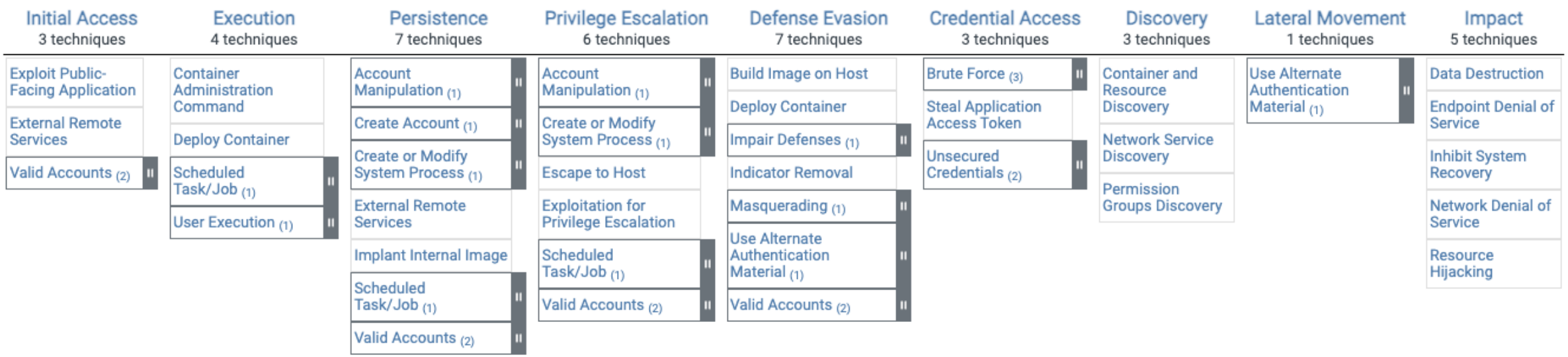
- R&D / Container Security в Luntry
- Специализируюсь на безопасности контейнеров и Kubernetes
- Багхантер
- Редактор telegram канала k8s (in)security
- Спикер: PHDays, OFFZONE, VK Kubernetes Conf, Devoops, HackConf, CyberCamp, БеКон и др.

План

- Матрицы атак
- Способы обнаружения
- Концентрируемся на Runtime
- Обнаружение
  - Продвинутое обнаружение
- Реакция

Матрицы атак

MITRE ATT&CK Container Matrix



MITRE ATT&CK Container Matrix – минусы

- Техники довольно абстрактны и не сильно погружены в контекст Kubernetes
- Самих техники сильно меньше по сравнению с другими матрицами
- Можно расценивать как инструмент, с помощью которого можно узнать об определенных процедурах

Threat Matrix for Kubernetes by Microsoft

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Impact
Using cloud credentials	Exec into container	Backdoor container	Privileged container	Clear container logs	List K8S secrets	Access Kubernetes API server	Access cloud resources	Images from a private registry	Data destruction
Compromised image In registry	bash/cmd inside container	Writable hostPath mount	Cluster-admin binding	Delete K8S events	Mount service principal	Access Kubelet API	Container service account	Collecting data from pod	Resource hijacking
Kubeconfig file	New container	Kubernetes CronJob	hostPath mount	Pod / container name similarity	Container service account	Network mapping	Cluster internal networking		Denial of service
Application vulnerability	Application exploit (RCE)	Malicious admission controller	Access cloud resources	Connect from proxy server	Application credentials in configuration files	Exposed sensitive interfaces	Application credentials in configuration files		
Exposed sensitive interfaces	SSH server running inside container	Container service account			Access managed identity credentials	Instance Metadata API	Writable hostPath mount		
	Sidecar injection	Static pods			Malicious admission controller		CoreDNS poisoning		
							ARP poisoning and IP spoofing		

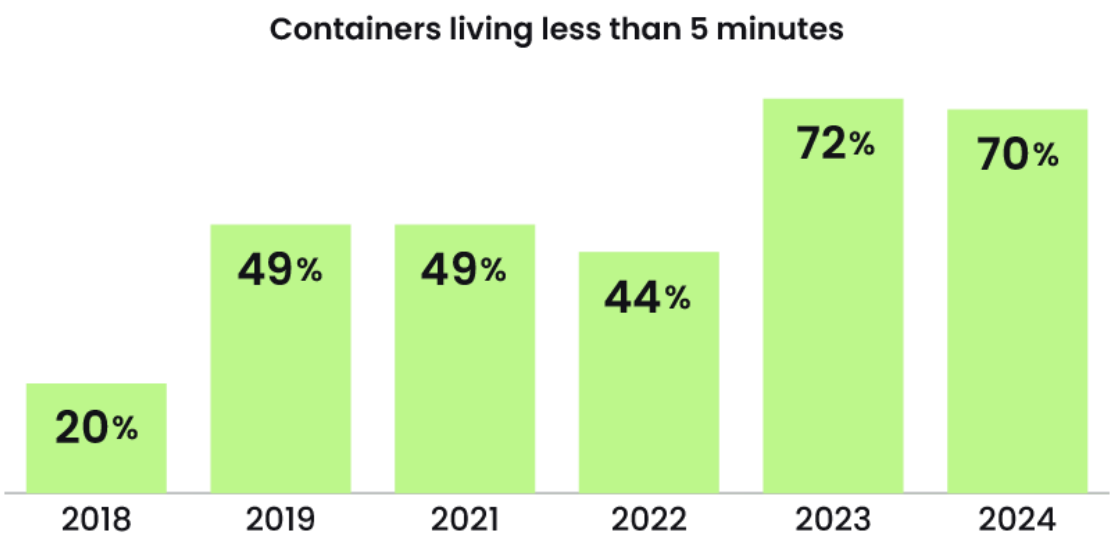
Threat Matrix for Kubernetes by Microsoft

- Понятные и хорошо описанные техники в контексте Kubernetes
- Есть маппинг на техники, описанные в MITRE
- Небольшой акцент на Managed K8s
- Как и в любой другой матрице, **атакующий всегда на шаг впереди**

VK Kubernetes Conf 2023. Экскурсия по матрицам угроз для контейнеров и Kubernetes

Способы обнаружения

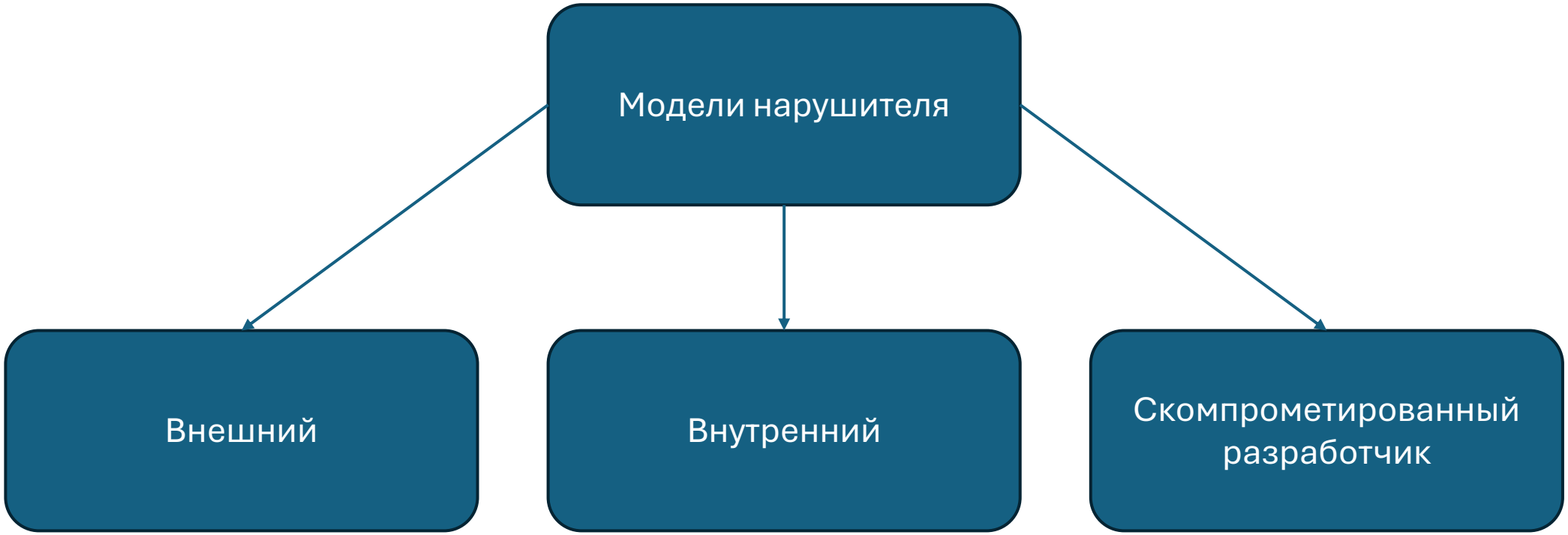
Динамическое окружение



Sysdig 2024 Cloud-Native Security and Usage Report

- Малый срок жизни контейнеров
- Self-healing
- Следы злоумышленника в контейнере очищаются сами собой

Модели нарушителя



KazHackStan 2022. Специфика расследования инцидентов в контейнерах(Дмитрий Евдокимов, Luntry)

Концентрируемся на Runtime

Виды защиты (Linux World)

- Isolation
  - Дополнительный уровень изоляции от ядра Host ОС (WASM, Sandbox, microVM, ...)
- Detection
  - Идентификация нежелательного действия
- Prevention
  - Невозможность выполнения нежелательного действия
- Mitigation
  - Смягчение последствий нежелательного действия
- Reaction
  - Ответ на нежелательное действие постфактум после нежелательного события

Сравнение возможностей Open Source решений

	Falco	Tracee	Tetragon
Базовая технология агента	eBPF, Kernel module	eBPF	eBPF
Обработка данных на user space	C\C++	Go	Go
Профилирование контейнеров	Нет	Нет	Нет
Отслеживаемые события	syscalls	syscalls, network, security, lsm, containers, misc	syscalls
Обработка событий	Client-side	Client-side	Client-side
Тип обнаружений	Сигнатурный (правила)	Сигнатурный (правила)	Сигнатурный (правила)
Создание политик/правил	Правила в ручную	Правила в ручную	Правила в ручную
Структура политики/правил	YAML	Rego/Go + YAML	YAML
Принцип работы	Blacklist	Blacklist	Blacklist
Привязка политик/правил	Правила в ручную	Правила в ручную	Правила в ручную
Принцип привязки политик/правил	Очень мощный фильтр	Scope	Namespace and pod label filtering (beta)
Режим работы	Detection	Detection	Detection, Reaction
Активное воздействие (reaction)	Нет	Нет	Завершение процесса
Расследование инцидента	Нет	Нет	Нет
Предотвращение процессных событий	Нет	Нет	Да!?*
Предотвращение сетевых событий	Нет	Нет	Да!?*

Runtime Security: на вкус и цвет все фломастеры разные (Дмитрий Евдокимов, Luntry)

Обнаружение


сигнатуры

Сигнатуры

```
- rule: Execution from /dev/shm
  desc: >
    This rule detects file execution in the /dev/shm directory, a tactic often used by threat actors to store th
    occasionally executable files. /dev/shm acts as a link to the host or other containers, creating vulnerabili
    as well. Notably, /dev/shm remains unchanged even after a container restart. Consider this rule alongside th
    "Drop and execute new binary in container" rule.
  condition: >
    spawned_process
    and (proc.exe startswith "/dev/shm/" or
        (proc.cwd startswith "/dev/shm/" and proc.exe startswith "./" ) or
        (shell_procs and proc.args startswith "-c /dev/shm") or
        (shell_procs and proc.args startswith "-i /dev/shm") or
        (shell_procs and proc.args startswith "/dev/shm") or
        (proc.cwd startswith "/dev/shm/" and proc.args startswith "./" ))
```

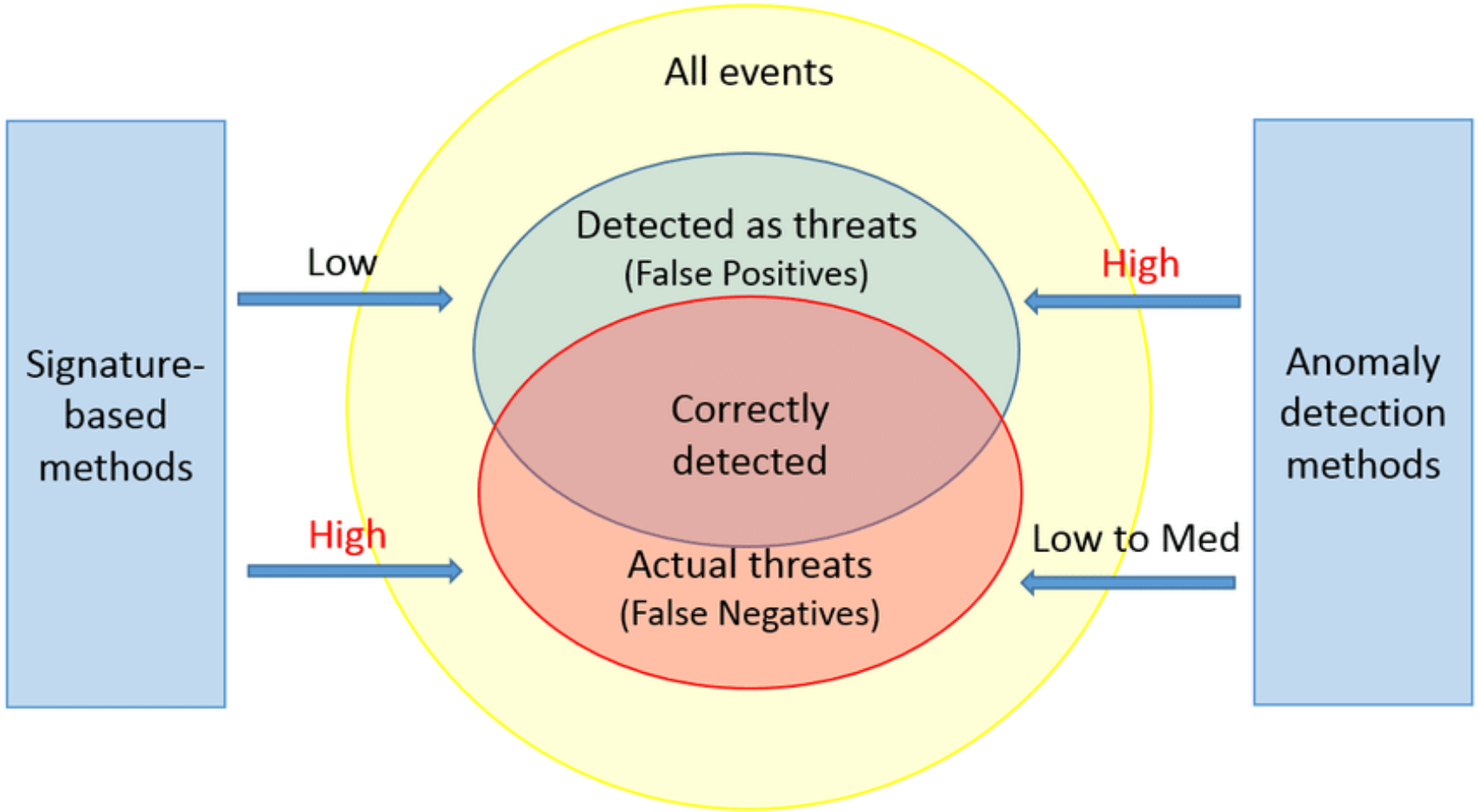
```
apiVersion: tracee.aquasec.com/v1beta1
kind: Policy
metadata:
  name: dig
  annotations:
    description: traces dns events from the dig executable
spec:
  scope:
    - executable=/usr/bin/dig
  rules:
    - event: net_packet_dns_request
    - event: net_packet_dns_response
```

```
apiVersion: cilium.io/v1alpha1
kind: TracingPolicy
metadata:
  name: "cve-2024-3094-xz-ssh"
  annotations:
    url: "https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2024-3094"
    description: "Detects if OpenSSH is using vulnerable XZ libraries"
    author: "Tetragon.io Team"
spec:
  kprobes:
  - call: "security_mmap_file"
    syscall: false
    return: true
    # message: "OpenSSH daemon using vulnerable XZ libraries CVE-2024-3094"
    # tags: [ "cve", "cve.2024.3094" ]
    args:
    - index: 0
      type: "file"
    - index: 1
      type: "uint32"
    - index: 2
      type: "nop"
    returnArg:
      index: 0
      type: "int"
    returnArgAction: "Post"
    selectors:
    - matchBinaries:
      - operator: "In"
        values:
        - "/usr/sbin/sshd"
      matchArgs:
      - index: 0
        operator: "Postfix"
        values:
        - "liblzma.so.5.6.0"
        - "liblzma.so.5.6.1"
      matchActions:
        - action: Post
          rateLimit: "1m"
```

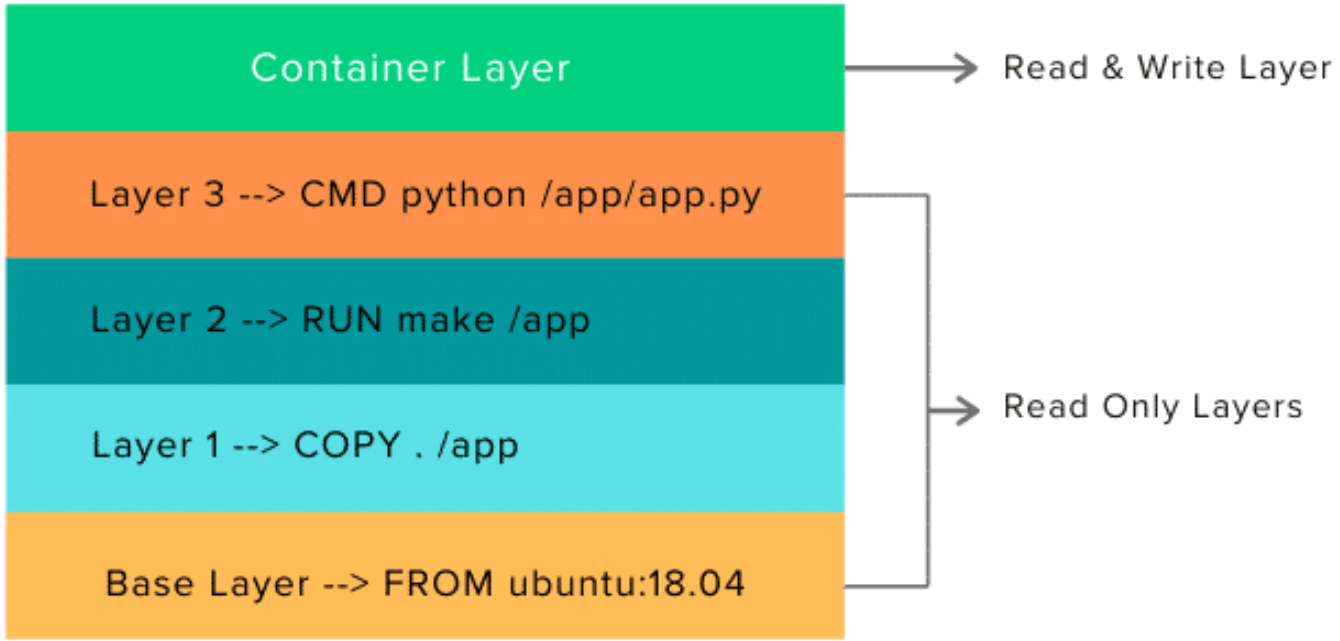
Проблема сигнатурного подхода

- Завязаны на определенное событие, syscall, filename, path, ...
- Уязвимости eBPF
- TOCTOU (Time-of-check time-of-use)
- Использование out of scope syscall (для средства защиты)
- eBPF map tampering
- Другие техники

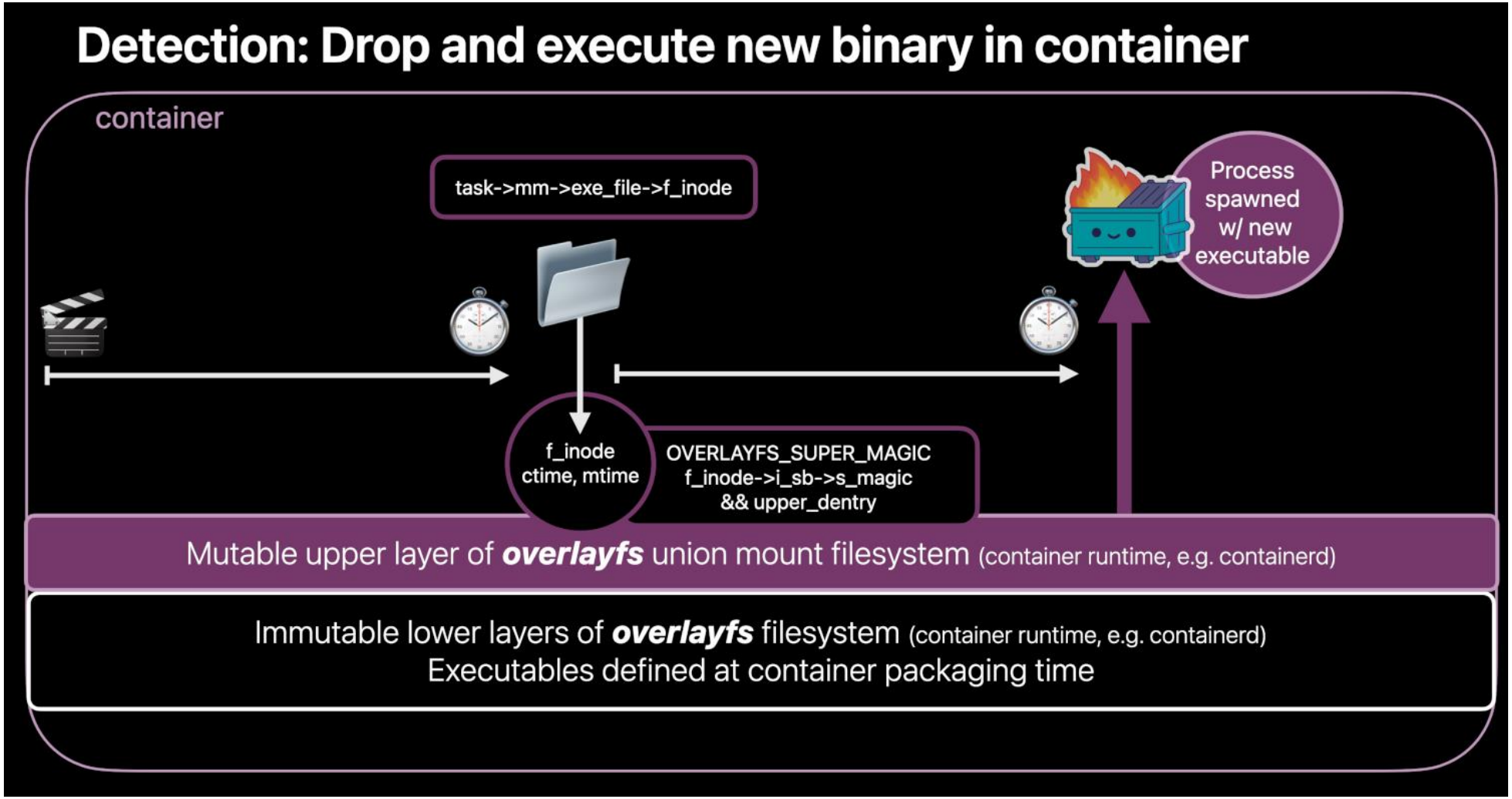
Signature Based VS Anomaly Based



Что такое OverlayFS?



Обнаружение новых бинарей в Upper Layer



Обход new binary execution [Falco]

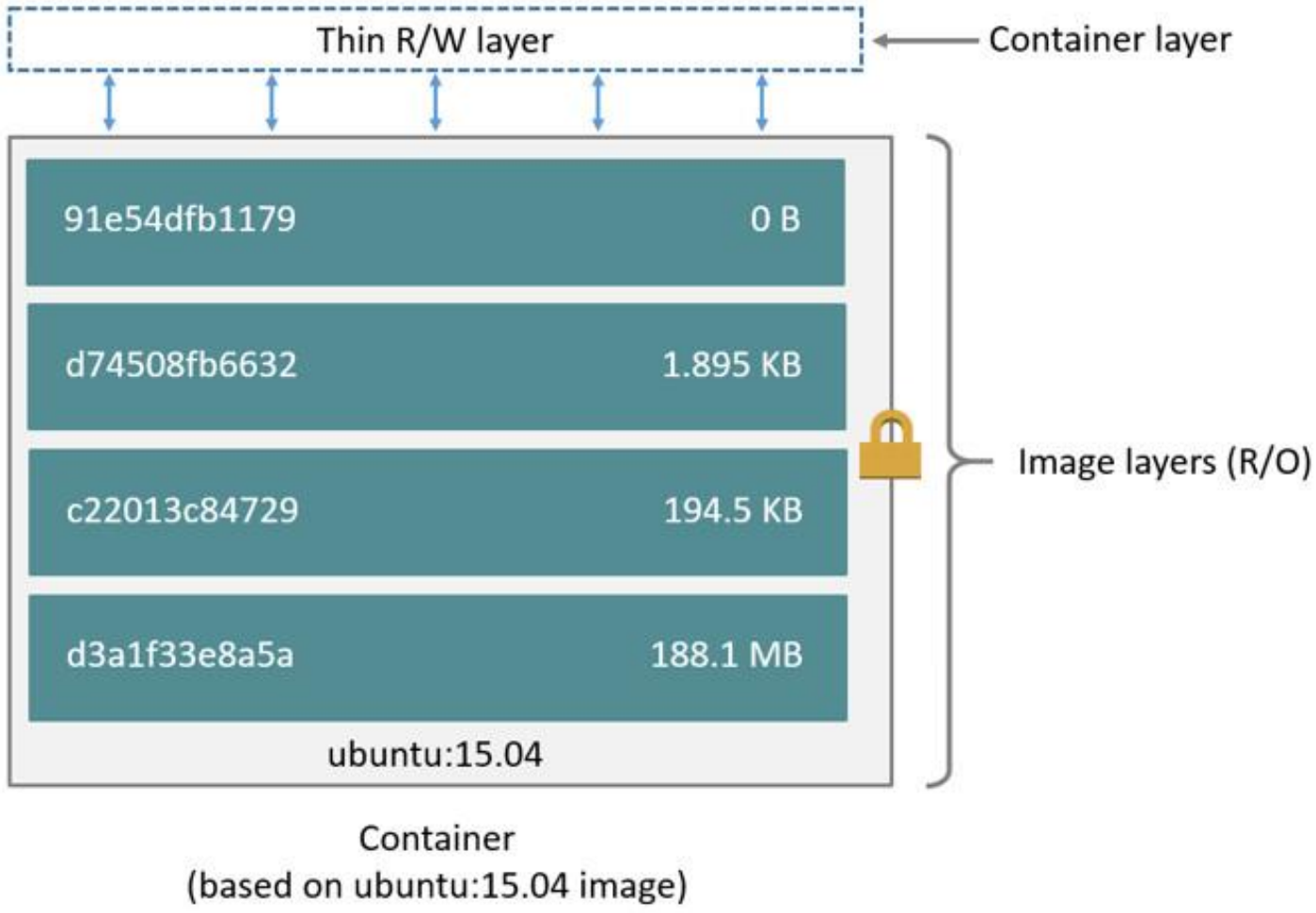
- Для обнаружения правило матчит событие по execve/execveat
- Чтобы обойти можно использовать GTFO bin, например ld.so
- Или воспользоваться техникой fileless execution

SOC-форум 2023. EDR vs Containers: актуальные проблемы (Владислав Лашкин, Solar; Дмитрий Евдокимов, Luntry)

Реакция

Дамп ФС

- Нет смысла дампать всю ФС целиком
- Нижние образы могут очень много весить
- Злоумышленник может взаимодействовать только с upper layer



Убийство контейнера

- Атакующий мог породить другие потоки, оставить для себя бэкдоры
- Такой контейнер это уже скомпрометированная среда

Заключение

- Классические подходы в контейнерах не эффективны
- Нужно использовать специфику контейнеров
- Сочетаний правил и аномалий