컨테이너 버그 찾기 어디까지 해봤니?

How To Find Container Platform Bug?

Team. MobyDick

구글의 모든 제품은 컨테이너에서 실행

→ 매주 수십억 개의 컨테이너 생성

80%

IT 조직 80% 이상이 컨테이너 사용 (직원 500명 이상)

Container

일관성 있는 환경

배포의 편의성

작은 이미지 크기

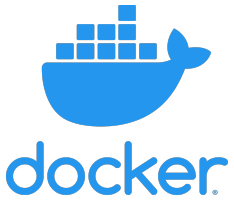
다양한 운영 환경 지원

주제 소개

“가트너, 컨테이너 보안을 올해 가장 우려되는 10가지 요소 중 하나로 선정”

“트렌드 마이크로, 기업의 86%이상이 쿠버네티스를 사용하며 컨테이너 보안에 대한 적절한 투자를 보유하고 있지 않아 쿠버네티스 채택이 빠르게 지연되거나 전략 없이 심각한 보안 사고가 발생”

주제 소개



Docker: 컨테이너 기반 가상화 플랫폼

Kubernetes: 컨테이너 오케스트레이션 도구

IT 조직 50% 이상이 Docker 사용, Kubernetes가 시장의 80% 이상 점유

컨테이너 버그 찾기 – Logic Bug?

CVE-2014-9357 - **wrong namespace**

CVE-2019-14271 - **wrong namespace**

CVE-2018-1002100 - **wrong namespace**

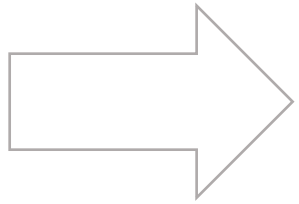
CVE-2019-1002101 - **wrong namespace**

CVE-2016-9962 - **File Descriptor**

CVE-2019-5736 - **File Descriptor**

CVE-2018-15664 - **Race Condition**

CVE-2019-19921 - **Race Condition**



컨테이너 버그 찾기 – Logic Bug?

CVE-2014-9357 - logic (Remote Code Execution)

CVE-2018-15514 - logic (Privilege Escalation)

CVE-2019-5736 - logic (Remote Code Execution)

CVE-2019-14271 - logic (Container Escape)

CVE-2019-15752 - logic (Privilege Escalation)

CVE-2016-1906 - logic (Privilege Escalation)

CVE-2018-1002101 - command injection (DoS)

CVE-2018-1002105 - logic (Privilege Escalation)

CVE-2019-11247 - logic (Arbitrary Read/Write)

CVE-2019-11248 - logic (DoS + Leak)

컨테이너 버그 찾기 – Logic Bug?

Docker

The underlying technology

Docker is written in the Go programming language and takes advantage of several features of the Linux kernel to deliver its functionality. Docker uses a technology called `namespaces` to provide the isolated workspace called the *container*. When you run a container, Docker creates a set of *namespaces* for that container.

These namespaces provide a layer of isolation. Each aspect of a container runs in a separate namespace and its access is limited to that namespace.

Kubernetes

The Kubernetes project is written in the Go programming language, and you can browse its source code on GitHub.

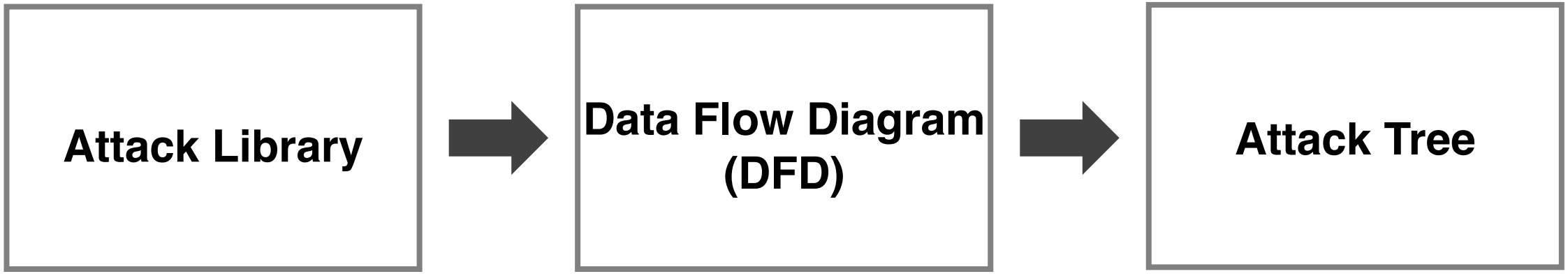
Go Language

Threat Modeling

CodeQL For GO (Code Auditing)

Introspection Tool

컨테이너 버그 찾기 – Threat Modeling



- 1-day Analysis
- Conference, journal, technical report, …

- Level 0, 1, 2

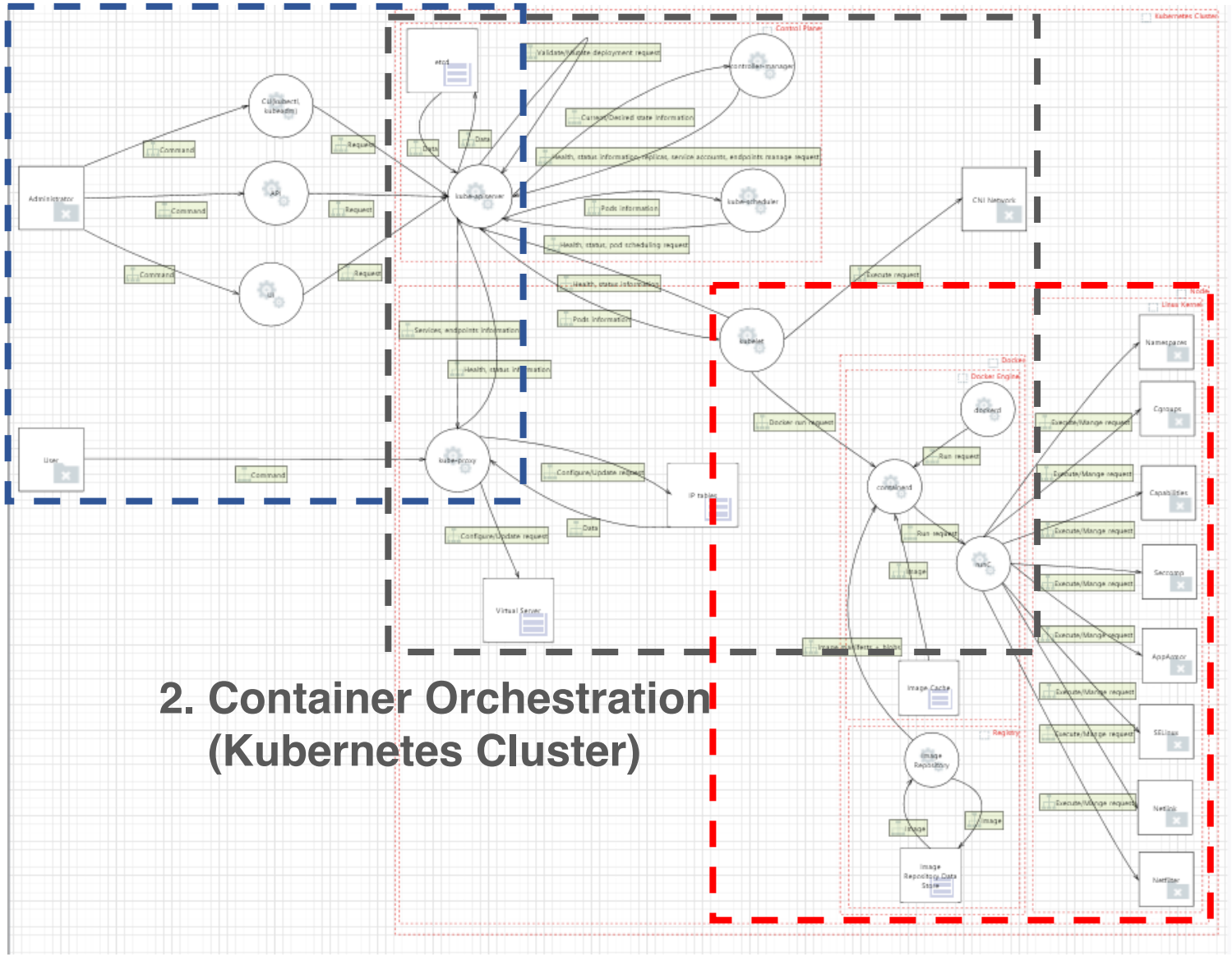
- DoS
- Info Leak
- Code Execution / Escape

컨테이너 버그 찾기 – Threat Modeling

Category	Title	Year	Author	File	Link
Conference	Escaping Virtualized Containers	2020	Yuval Avrahami	us-20-A...	https://www.blackhat.com/us-20/briefings/schedule/#escaping-virtualized-containers-20514
Conference	A Compendium of Container Escapes	2019	Brandon Edwards, Nick Freeman	us-19-E...	https://www.blackhat.com/us-19/briefings/schedule/#a-compendium-of-container-escapes-16091
Conference	Docker Escape Technology	2016	Shengping Wang	CSW20...	https://cansecwest.com/slides/2016/CSW2016_Wang_DockerEscapeTechnology.pdf
Conference	Abusing Privileged and Unprivileged Linux Containers	2016	Jesse Hertz	contain...	https://www.nccgroup.com/uk/our-research/abusing-privileged-and-unprivileged-linux-containers/
Conference	VULNERABILITY EXPLOITATION IN DOCKER CONTAINER ENVIRONMENTS	2015	Anthony Bettini	eu-15-B...	https://www.blackhat.com/eu-15/briefings.html#vulnerability-exploitation-in-docker-container-environments
Journal	Vulnerability Analysis and Security Research of Docker Container	2020	Jiang Wenhao, Li Zheng	092368...	https://ieeexplore.ieee.org/abstract/document/9236837
Journal	Analysis of Docker Security	2015	Thanh Bui	1501.02...	https://arxiv.org/abs/1501.02967
Journal	To Docker or Not to Docker: A Security Perspective	2016	Theo Combe, Antony Martin, Roberto Di Pietro	077422...	https://ieeexplore.ieee.org/abstract/document/7742298
Journal	Docker container security via heuristics-based multilateral security-conceptual and pragmatic study	2016	A R Manu, Jitendra Kumar Patel, Shakil Akhtar, V K Agrawal, K N Bala Subramanya Murthy	075302...	https://ieeexplore.ieee.org/abstract/document/7530217
Journal	Docker ecosystem – Vulnerability Analysis	2018	A. Martin, S. Raponib, T. Combea, R. Di Pietro	1-s2.0-...	https://www.sciencedirect.com/science/article/abs/pii/S0140366417300956
Journal	Securing Docker Containers from Denial of Service (DoS) Attacks	2016	Jeeva Chelladhurai, Pethuru Raj Chelliah, Sathish Alampalayam Kumar	075575...	https://ieeexplore.ieee.org/abstract/document/7557545
Technical Report	Guide to Container Security – Everything You Need to Know	2019	Michelle Moore		https://www.tripwire.com/state-of-security/devops/guide-container-security/
White Paper	Docker security	2020	Docker, Inc.		https://docs.docker.com/engine/security/
White Paper	LXC security	2020	Canonical Ltd.		https://linuxcontainers.org/lxc/security/
White Paper	Kubernetes security	2020	Cloud Native Computing Foundation		https://kubernetes.io/docs/concepts/security/
Public Program or Project	Kubernetes Security - Best Practice Guide	2018	Simon Pirschel		https://github.com/freach/kubernetes-security-best-practice

컨테이너 버그 찾기 – Threat Modeling

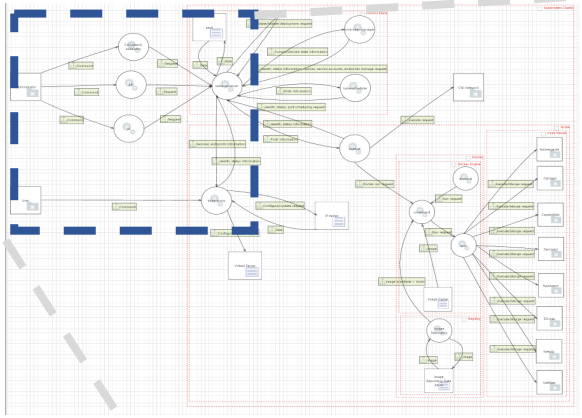
1. User Interaction



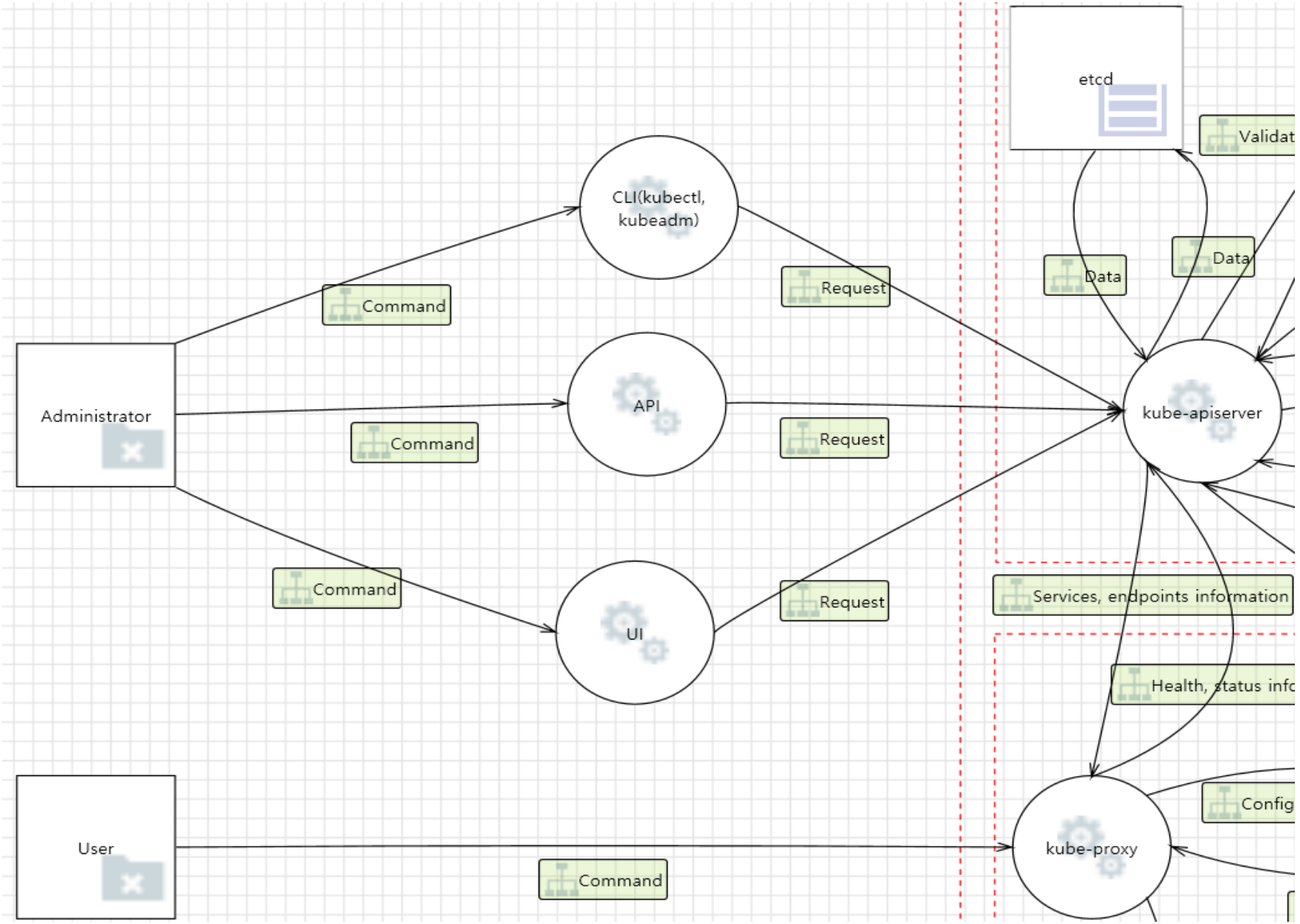
2. Container Orchestration
(Kubernetes Cluster)

3. Container Runtime
(Docker & Kernel)

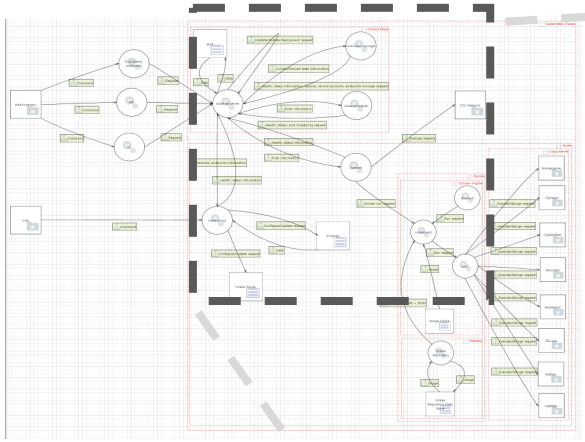
컨테이너 버그 찾기 – Threat Modeling



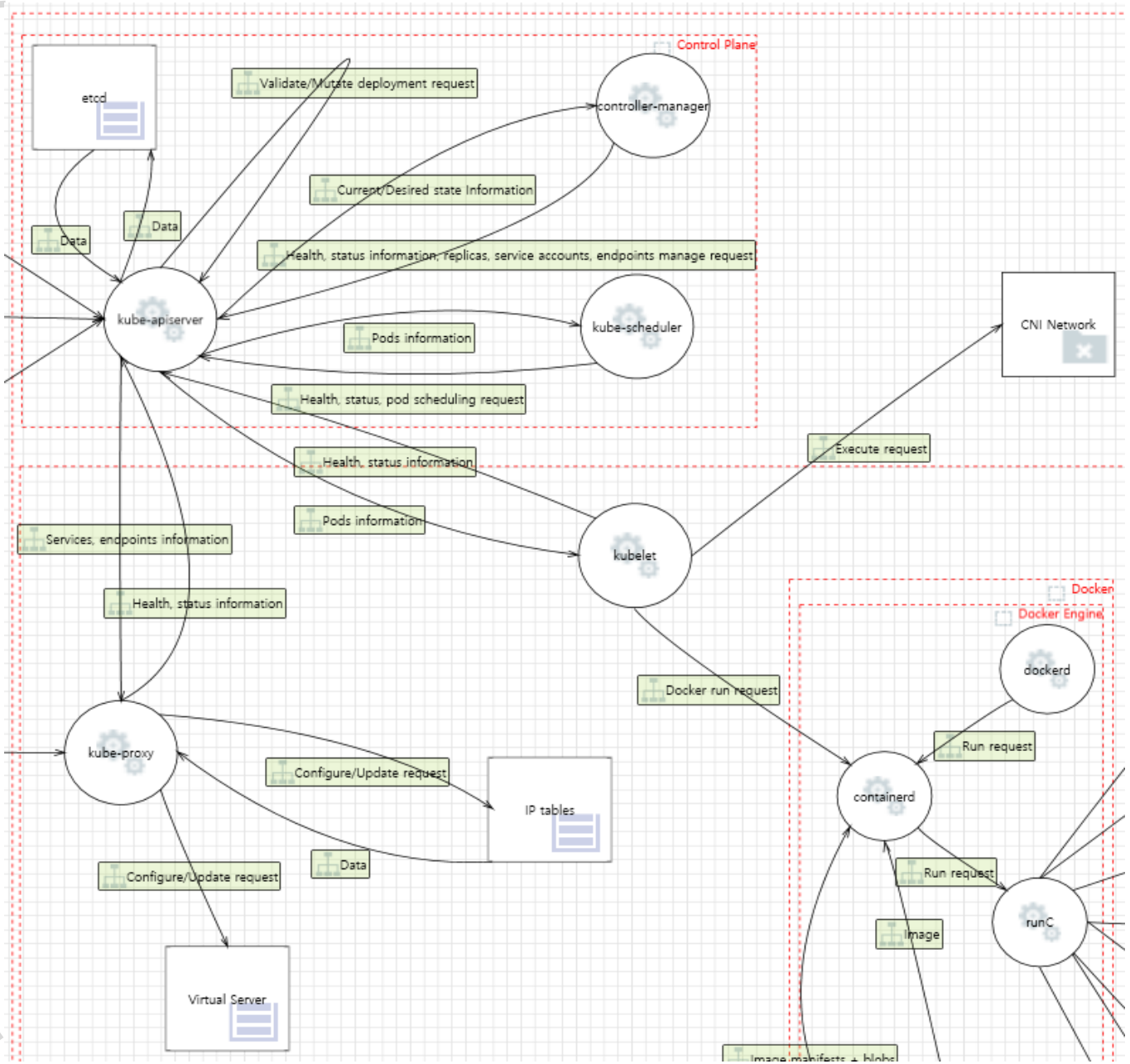
1. User Interaction



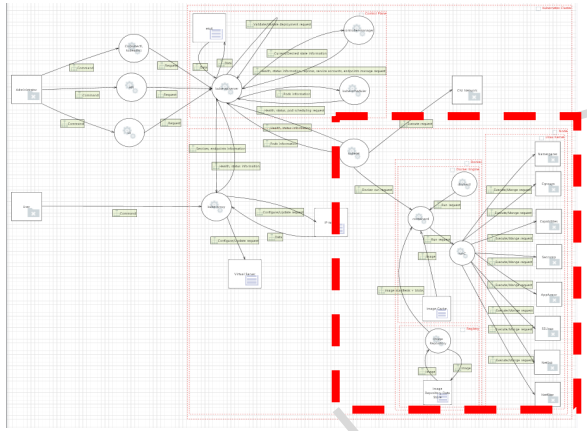
컨테이너 버그 찾기 – Threat Modeling



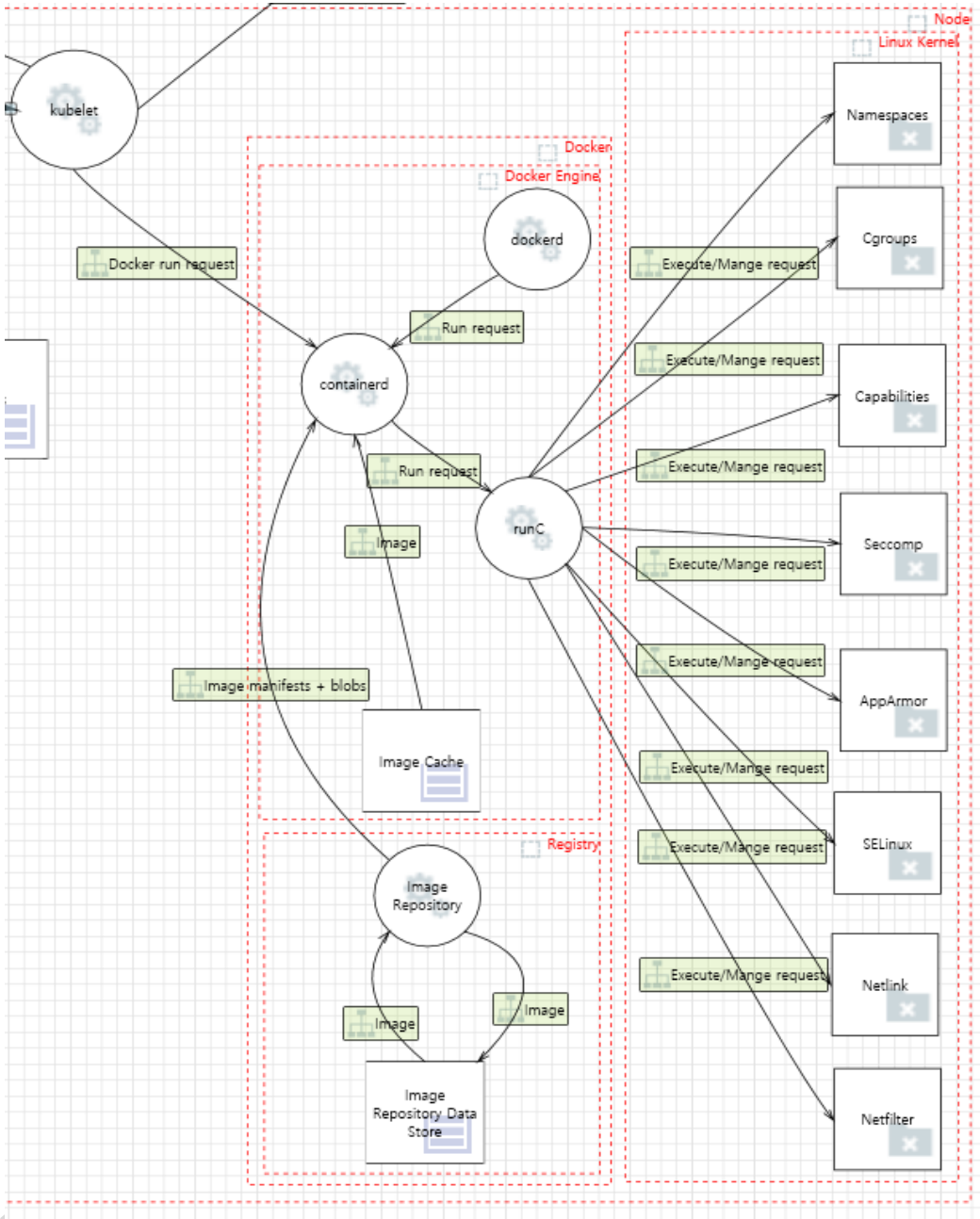
2. Container Orchestration (Kubernetes Cluster)



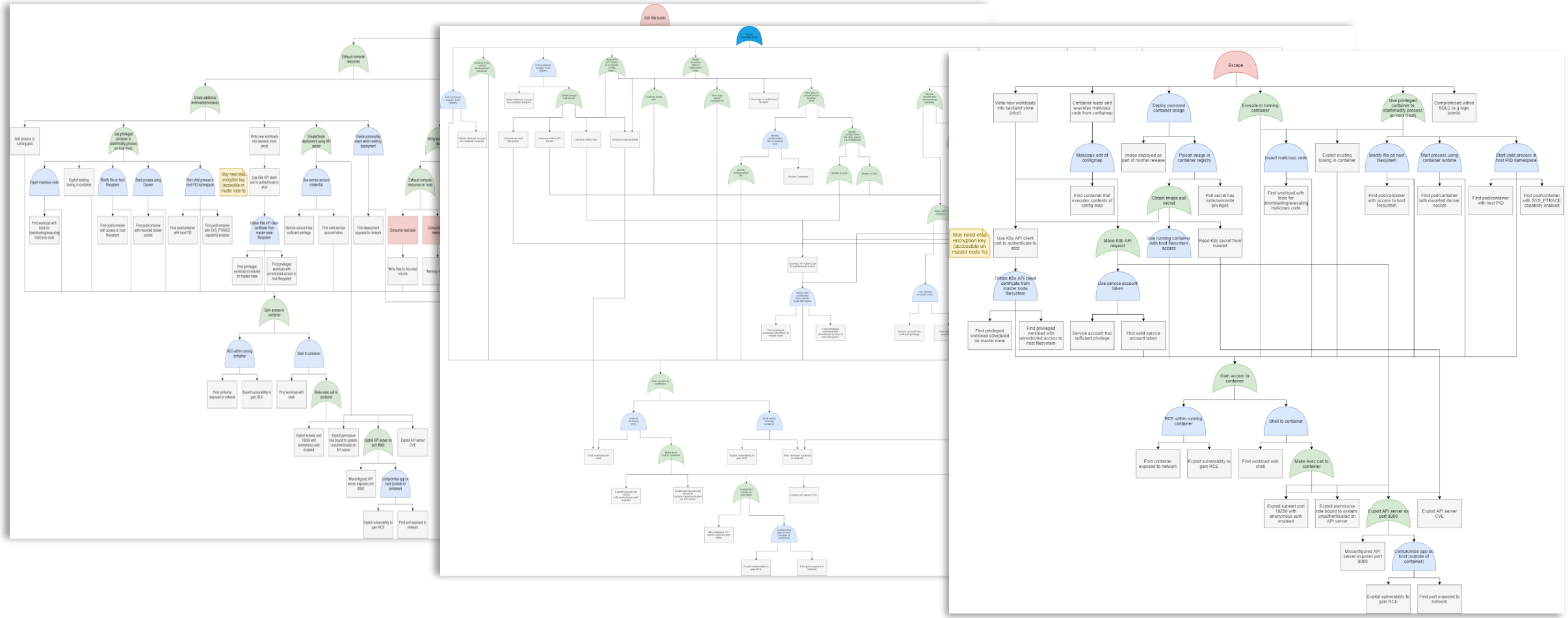
컨테이너 버그 찾기 – Threat Modeling



3. Container Runtime (Docker & Kernel)



컨테이너 버그 찾기 – Threat Modeling

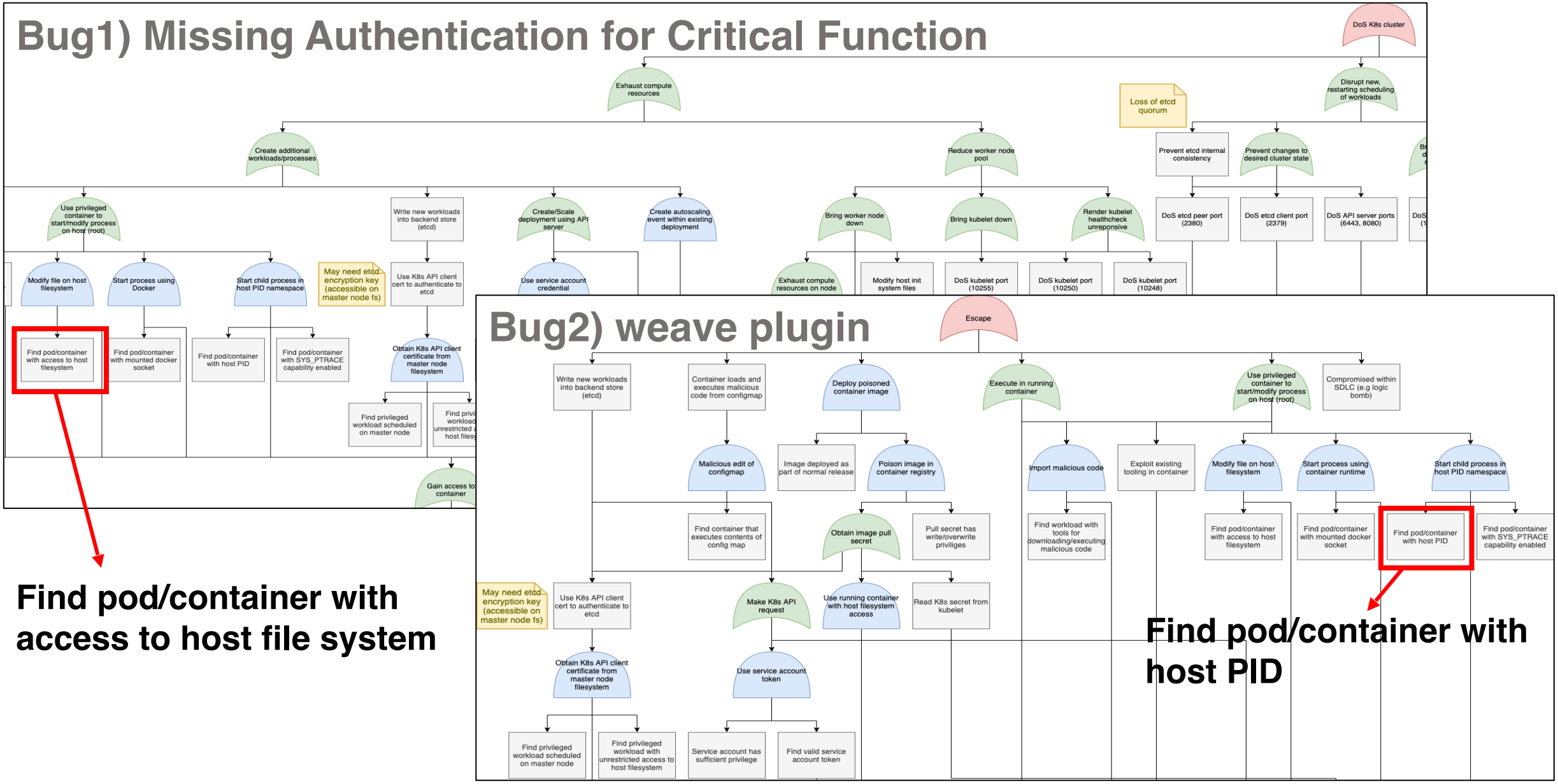


DoS

Information Leak

Code Execution & Escape

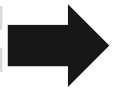
컨테이너 버그 찾기 – Threat Modeling

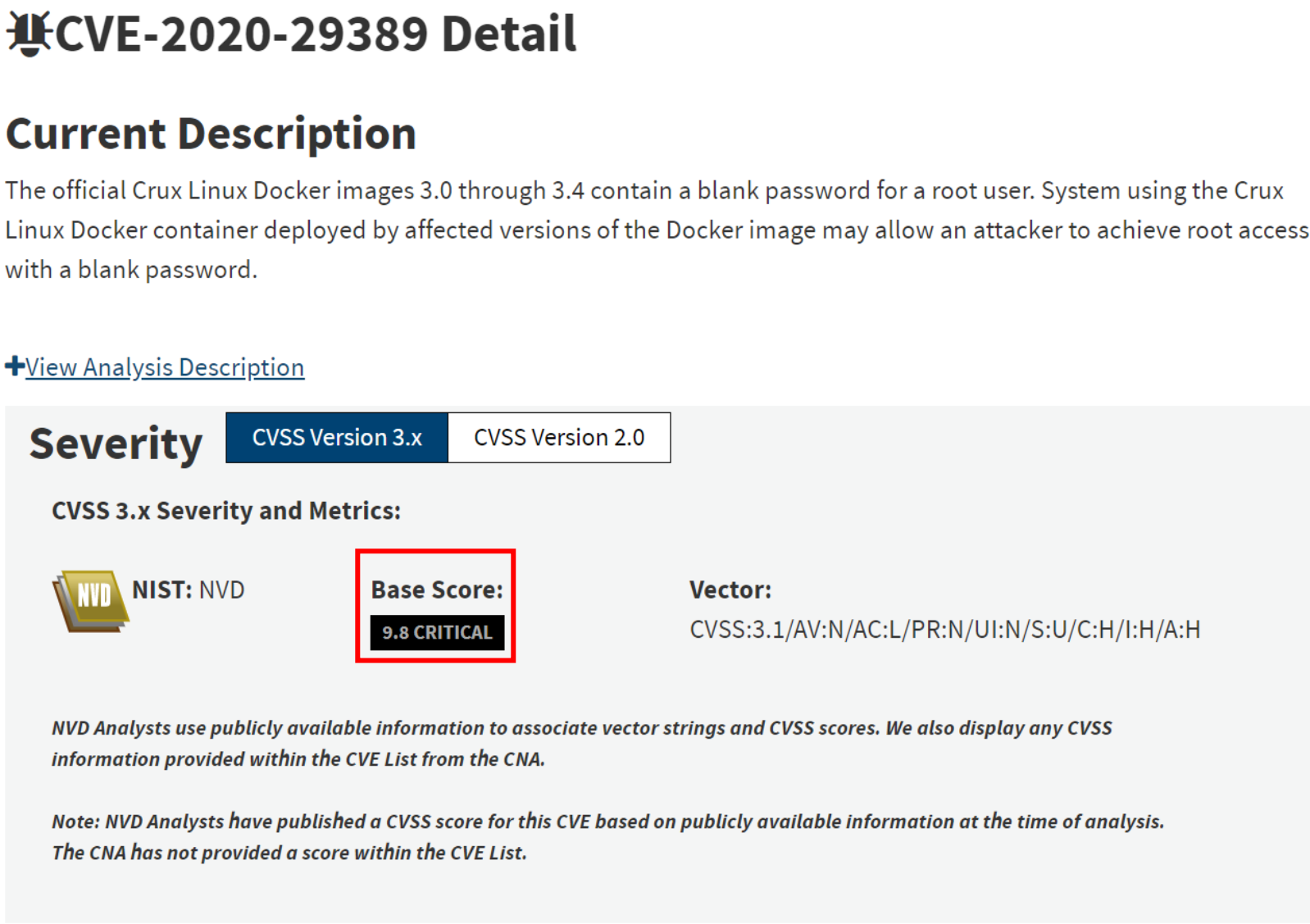


컨테이너 버그 찾기 – Threat Modeling
Missing Authentication for Critical Function

- 컨테이너에 **잘못된 권한 설정**
  - Root / Non-Root
- Remote Vulnerability
- **CVSS Score: 9.8**

One of the vulnerabilities we found





CVE-2020-29389 Detail

Current Description

The official Crux Linux Docker images 3.0 through 3.4 contain a blank password for a root user. System using the Crux Linux Docker container deployed by affected versions of the Docker image may allow an attacker to achieve root access with a blank password.

+View Analysis Description

Severity

CVSS Version 3.x | CVSS Version 2.0

CVSS 3.x Severity and Metrics:

NIST: NVD

Base Score: 9.8 CRITICAL

Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

NVD Analysts use publicly available information to associate vector strings and CVSS scores. We also display any CVSS information provided within the CVE List from the CNA.

Note: NVD Analysts have published a CVSS score for this CVE based on publicly available information at the time of analysis. The CNA has not provided a score within the CVE List.

- CodeQL
  - 코드 DB화, 쿼리로 원하는 패턴 검색
  - Java, C++, C#, Python, Go 등 다양한 언어 지원
- CodeQL for ‘Go’
  - **2019.12 first release**
  - 다른 언어보다 지원 문법↓, 레퍼런스↓

컨테이너 버그 찾기 – CodeQL For Go (Code Auditing)

1-day CodeQL

cmd/kubelet/app/server.go

```
@@ -746,9 +746,10 @@ func run(s *options.KubeletServer, kubeDeps *kubelet.Dependencies, stopCh <-chan
        }

        if s.HealthzPort > 0 {
-               healthz.DefaultHealthz()
+               mux := http.NewServeMux()
+               healthz.InstallHandler(mux)
                go wait.Until(func() {
-                       err := http.ListenAndServe(net.JoinHostPort(s.HealthzBindAddress, strconv.Itoa(int(s.HealthzPort))), nil)
+                       err := http.ListenAndServe(net.JoinHostPort(s.HealthzBindAddress, strconv.Itoa(int(s.HealthzPort))), mux)
                        if err != nil {
                                glog.Errorf("Starting health server failed: %v", err)
                        }
```

```
import go

from Function listenandserve, DataFlow::CallNode call, DataFlow::Node argument, DataFlow::Node nil
where
  listenandserve.hasQualifiedName("net/http", "ListenAndServe") and
  call = listenandserve.getACall() and
  argument = call.getArgument(1) and
  nil = Builtin::nil().getARead() and
  argument = nil
select call, call.getFile()
```

CodeQL 분석

1-day 바탕으로 CodeQL query 작성

- **zip-slip.ql**

→ Kubernetes CVE-2019-1002100

- **bad-coding-style.ql**

→ Kubernetes CVE-2019-11248

- **xpath-injection.ql**

→ Docker 19.03.13

컨테이너 버그 찾기 – CodeQL For Go (Code Auditing)

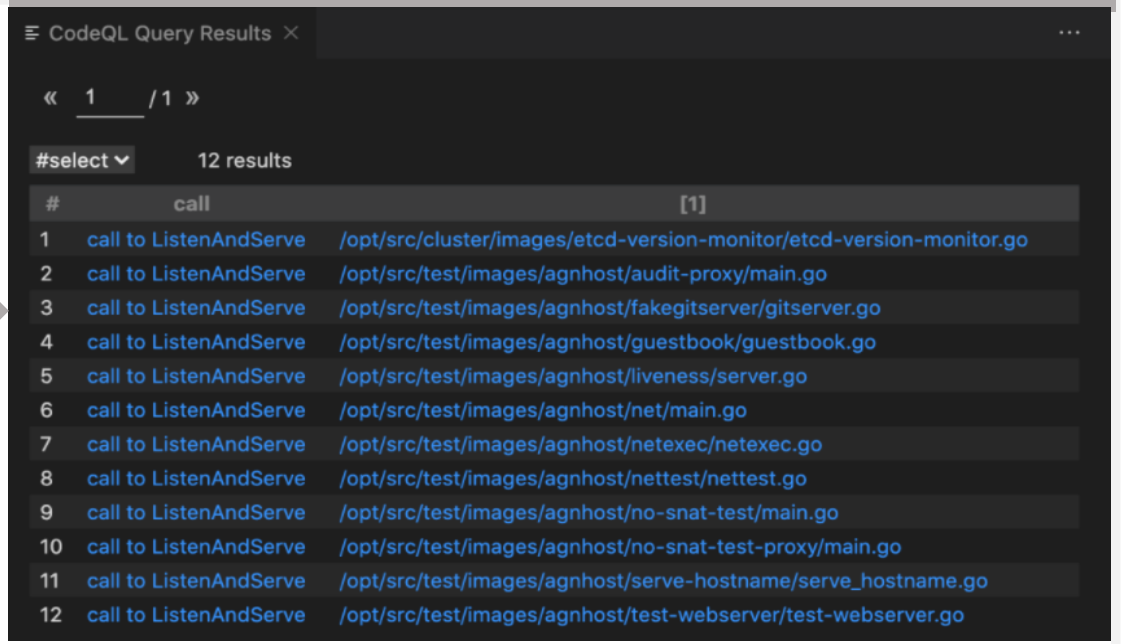
1-day CodeQL

```
/**
 * @name Unreachable statement
 * @description Unreachable statements are often indicative of missing code or latent bugs
 *              and should be avoided.
 * @kind problem
 * @problem.severity warning
 * @id go/unreachable-statement
 * @tags maintainability
 *       correctness
 *       external/cwe/cwe-561
 * @precision very-high
 */

import go

ControlFlow::Node nonGuardPredecessor(ControlFlow::Node nd) {
  exists(ControlFlow::Node pred | pred = nd.getAPredecessor() |
    if pred instanceof ControlFlow::ConditionGuardNode
    then result = nonGuardPredecessor(pred)
    else result = pred
  )
}
```

CodeQL 분석



Query를 최신 Docker, Kubernetes 버전에 적용

컨테이너 버그 찾기 – CodeQL For Go (Code Auditing)

CodeQL for Go & Docker 소스코드 개선

CodeQL for Go Github에 issue 작성

Feature request: specify Go version used to build project #351

Open donghyunlee00 opened this issue 19 days ago · 4 comments

donghyunlee00 commented 19 days ago

Hello.

I tried to create a CodeQL database for kubernetes-1.9.0 (which is an old version) through LGTM.com.

It was built successfully, but I found a warning in the Golang 'Extraction' log, as follows.

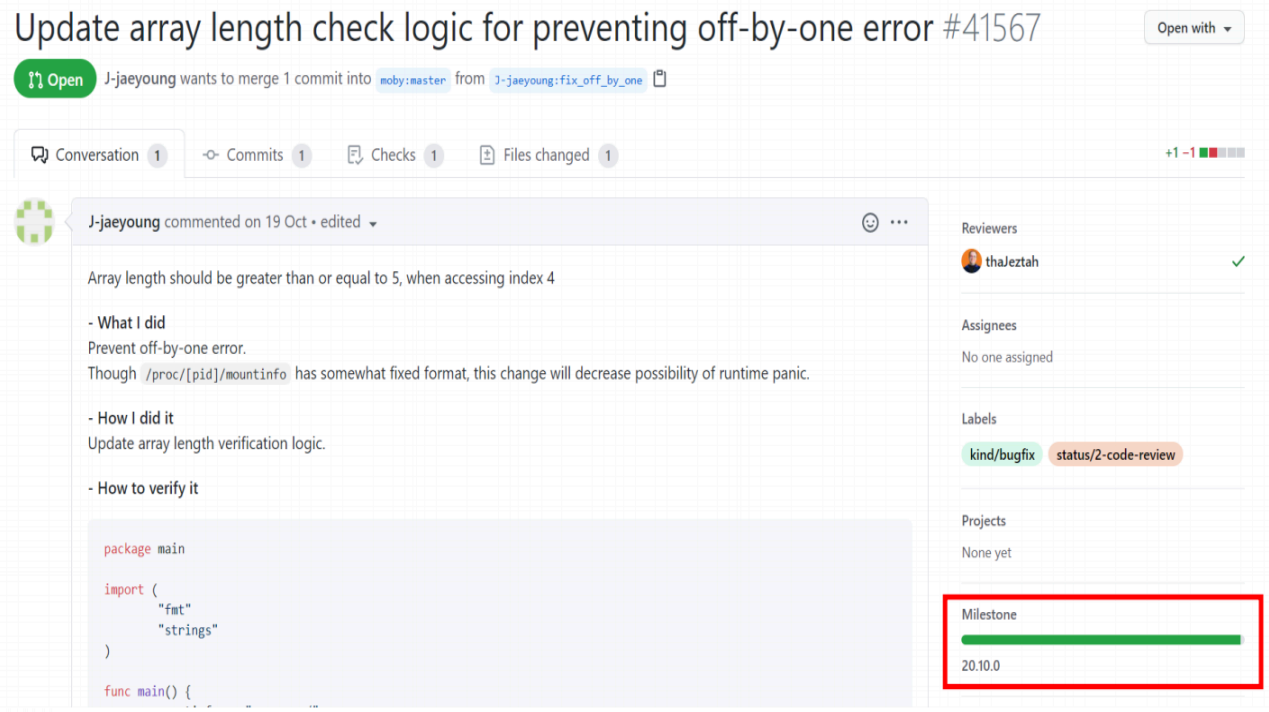
```
...
[build] Detected go version: go version go1.15 linux/amd64.
[build] Kubernetes requires go1.9.1 or greater.
[build] Please install go1.9.1 or later.
...
```

As far as I know, I can specify the version through `lgtm.yml` for languages such as Python, C#, Java.

However, in this document, I could not find any words to specify the version of Golang.

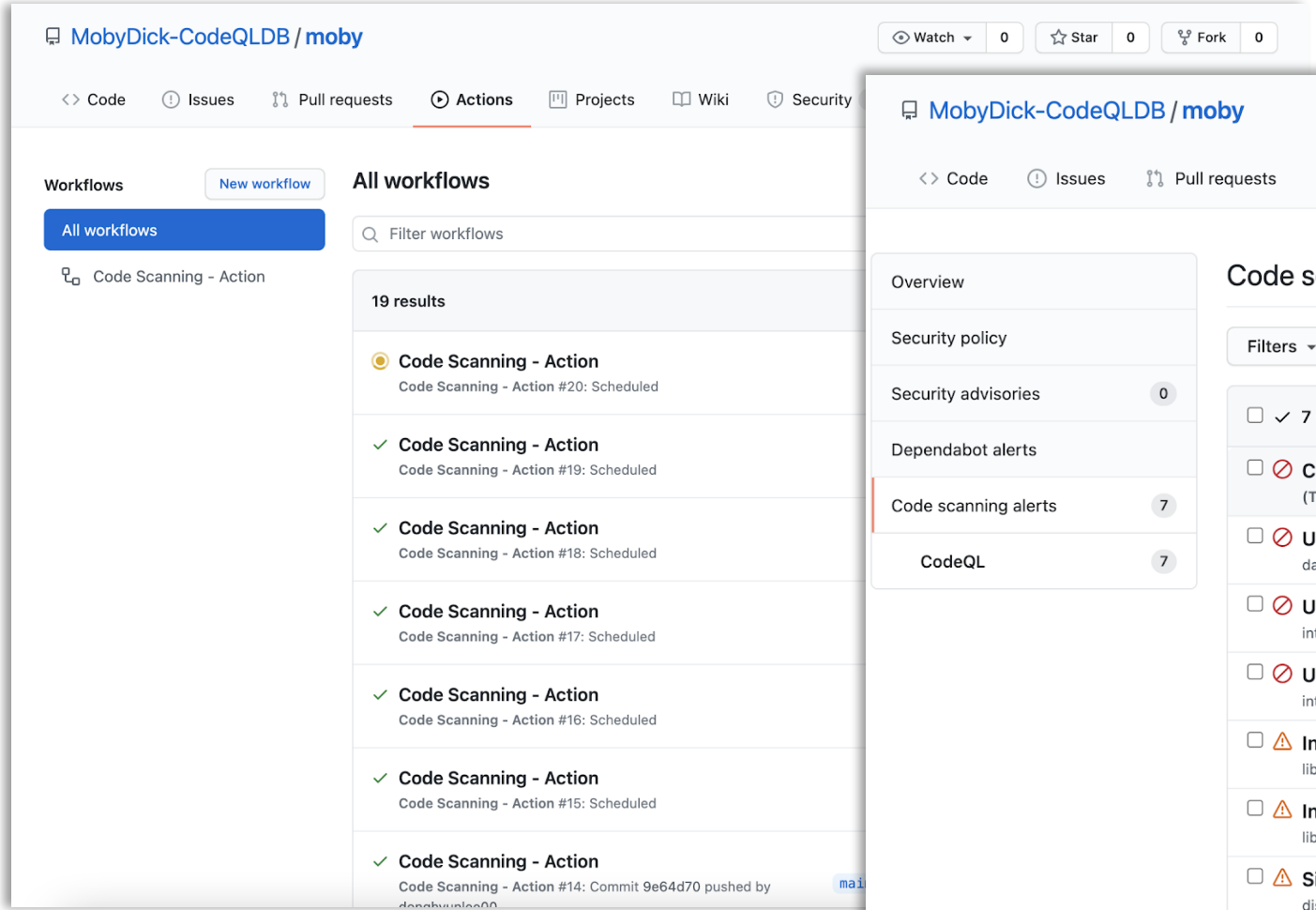
Is there any way to specify the Golang version when creating a CodeQL database?

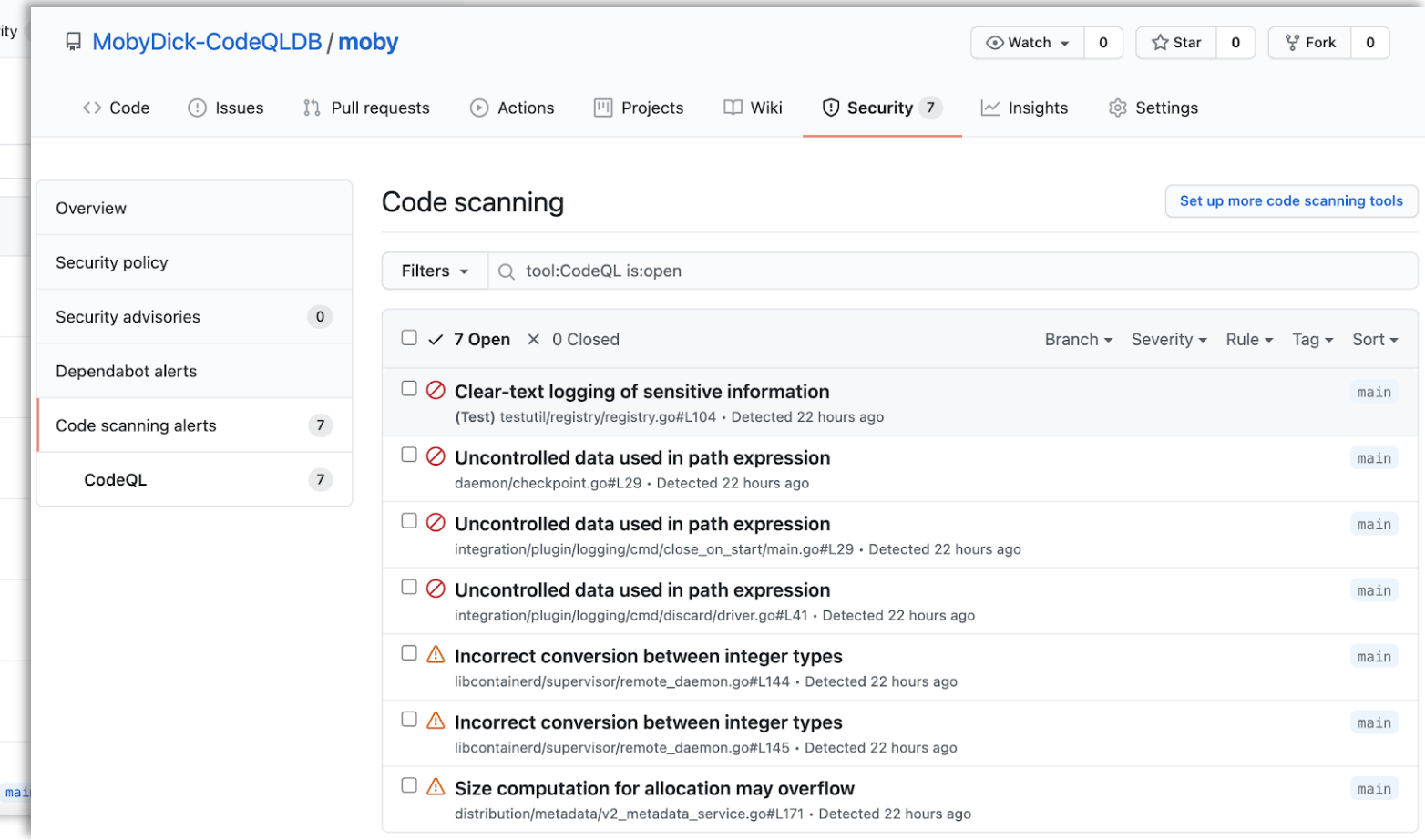
Docker Github에 pull request 작성



컨테이너 버그 찾기 – CodeQL For Go (Code Auditing)

CodeQL Action

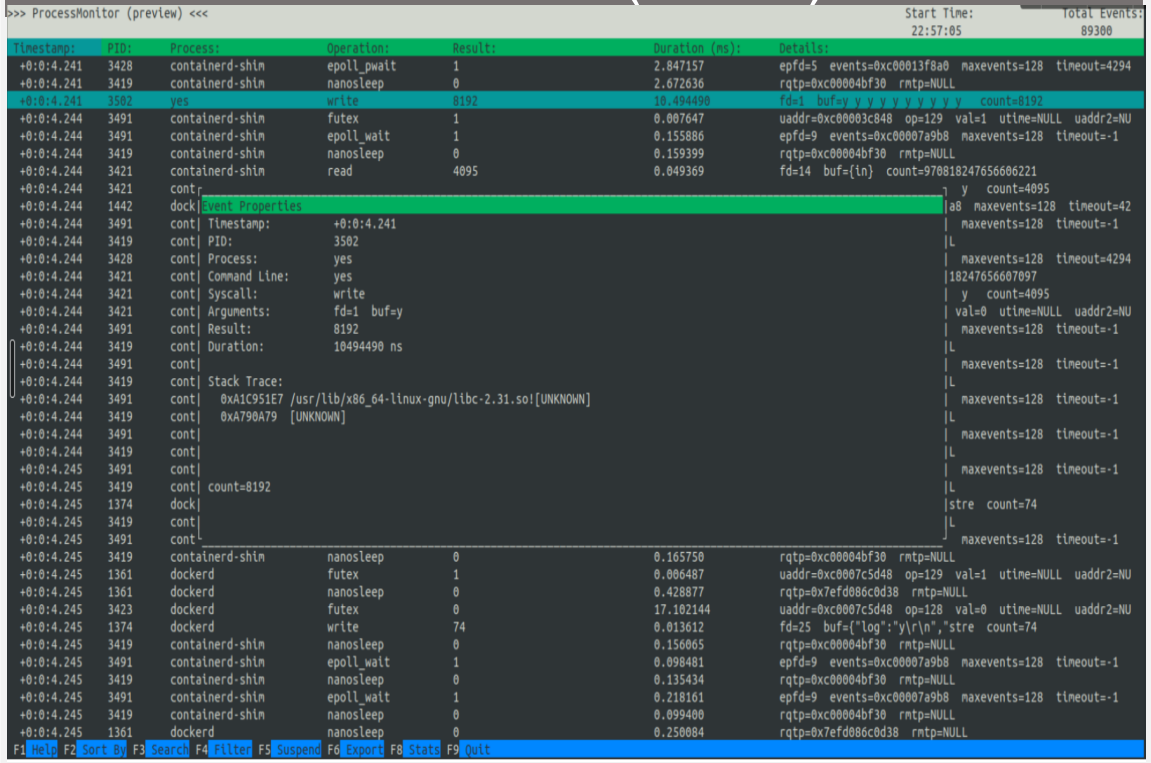




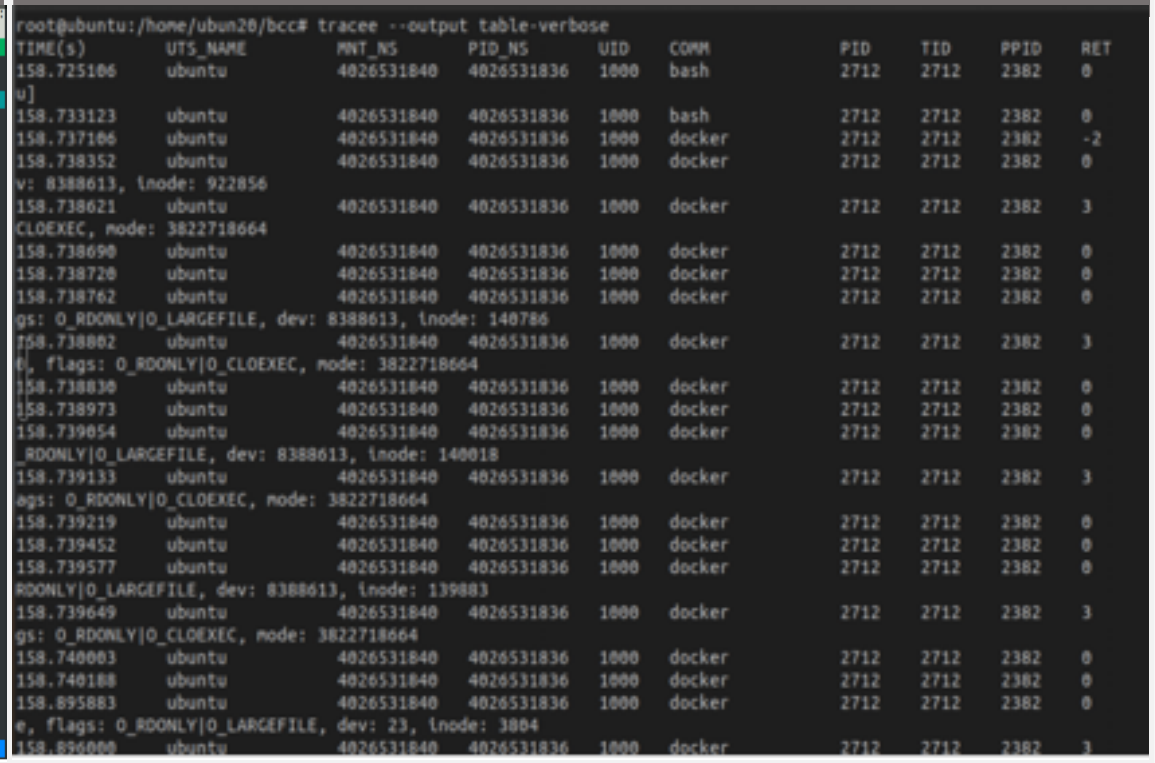
컨테이너 버그 찾기 – Introspection Tool

“ 도커 사용 시 호스트와 컨테이너가 분리,
모니터링하는 입장에서는 호스트와 컨테이너를 **구분**할 수 있어야 함 “

ProcMon (Linux)



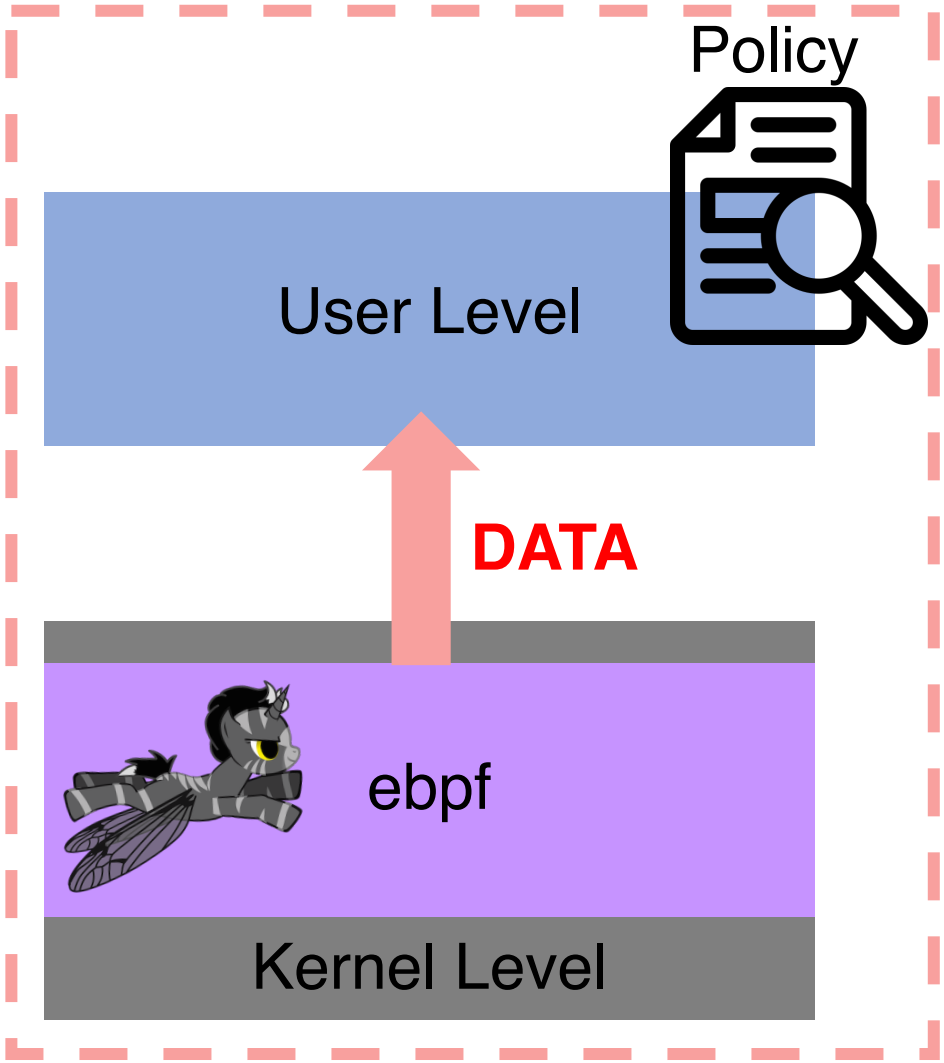
Tracee



컨테이너 버그 찾기 – Introspection Tool

- 컨테이너 기본 정보
- 프로세스 생성
  - 사용하는 **바이너리의 전체 경로** 및 **Namespace** 관찰
  - 프로세스가 **동적으로 로딩하는 라이브러리의 전체 경로 확인**
- 프로세스의 **Capability**
- **정책 위반 탐지**
  - 호스트의 namespace 로 **컨테이너의 라이브러리를 로딩하지 않아야 함**
  - 호스트의 namespace 로 **컨테이너의 바이너리를 사용하지 않아야 함**
  - 컨테이너가 호스트에 대한 **파일 식별자를 보유하지 않아야 함**

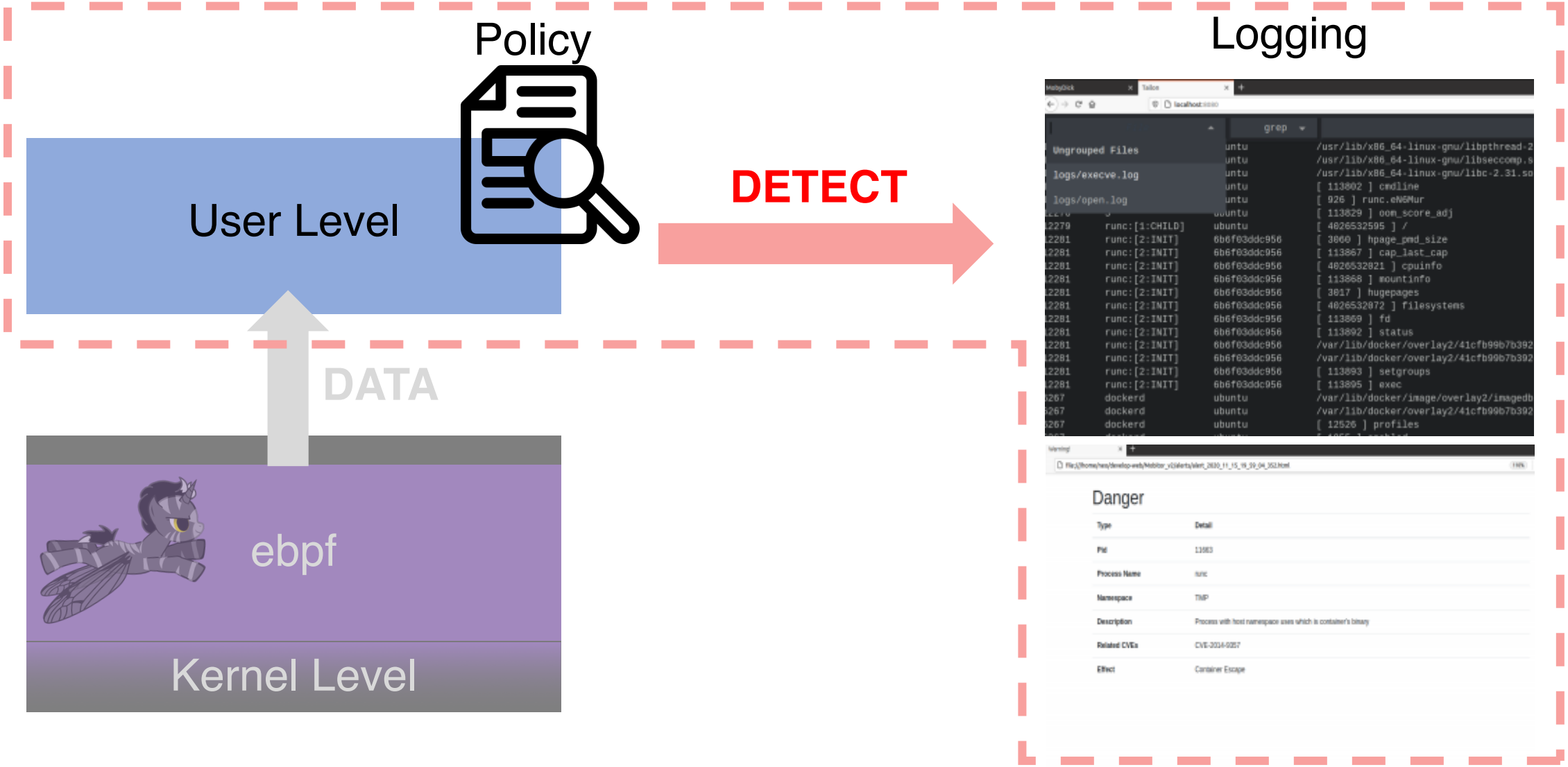
컨테이너 버그 찾기 – Introspection Tool



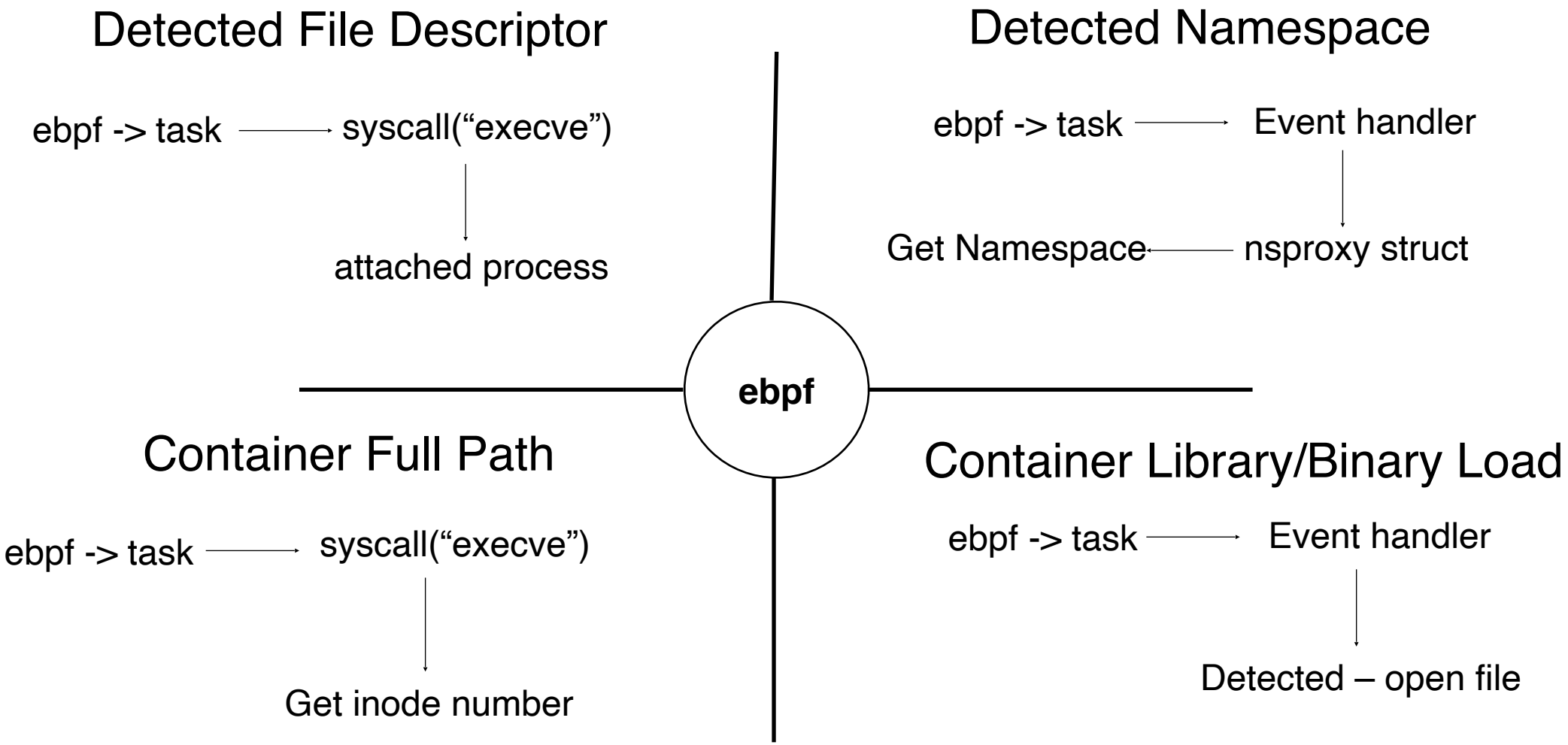
DETECT

Logging

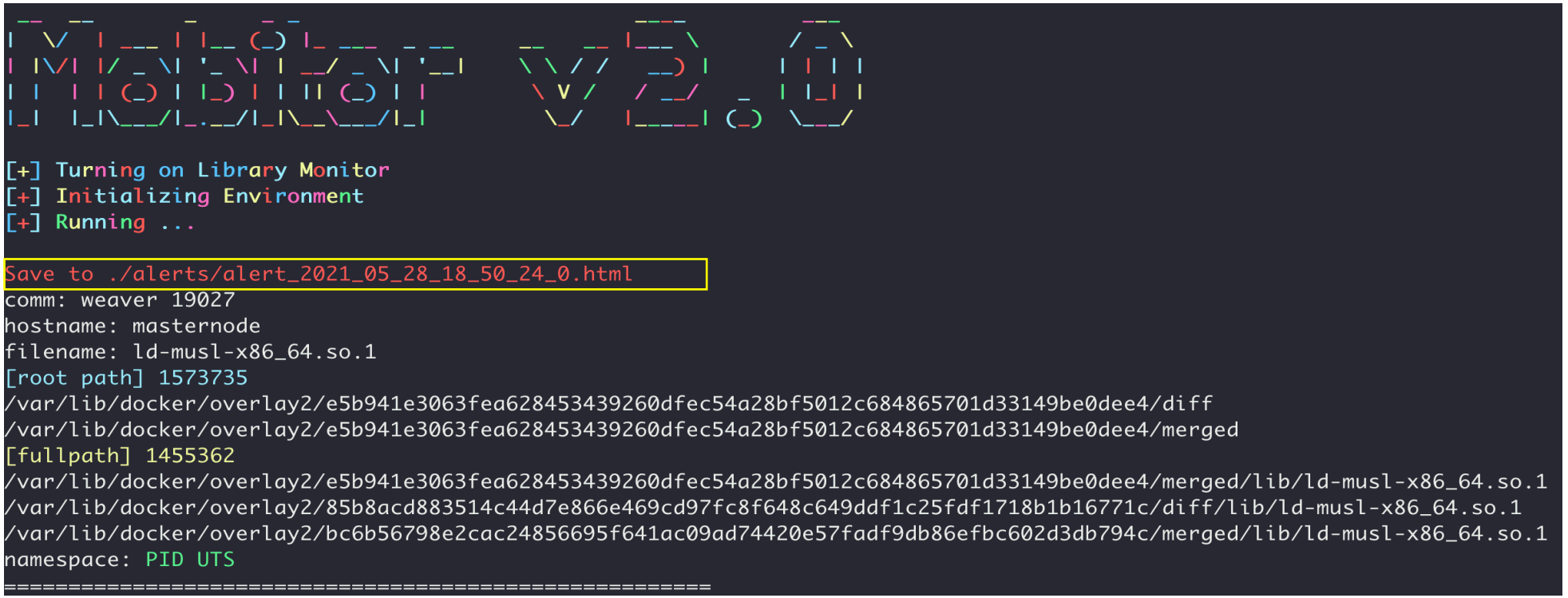
컨테이너 버그 찾기 – Introspection Tool



컨테이너 버그 찾기 – Introspection Tool



컨테이너 버그 찾기 – Introspection Tool

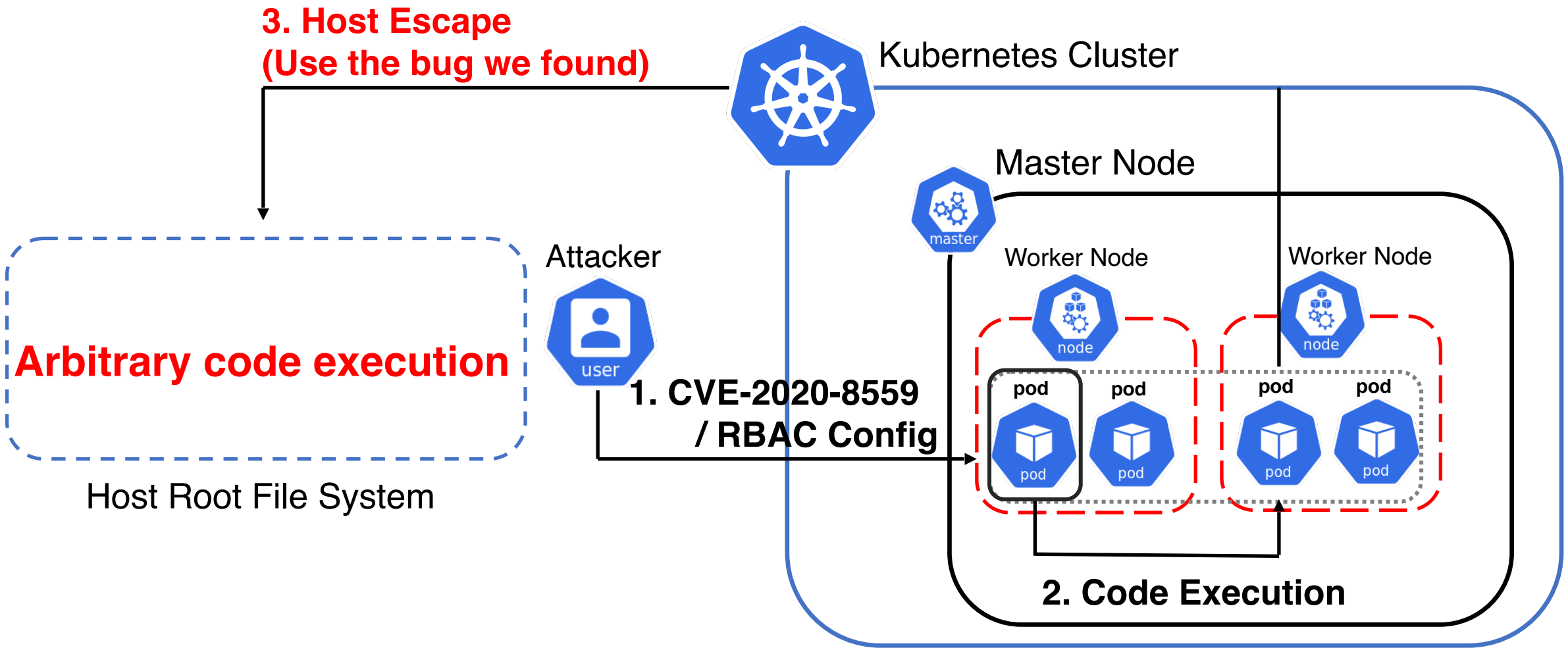


- Docker Full Escape Bug **4개 중 3개** 탐지
- Kubernetes Escape Bug **2개** 탐지

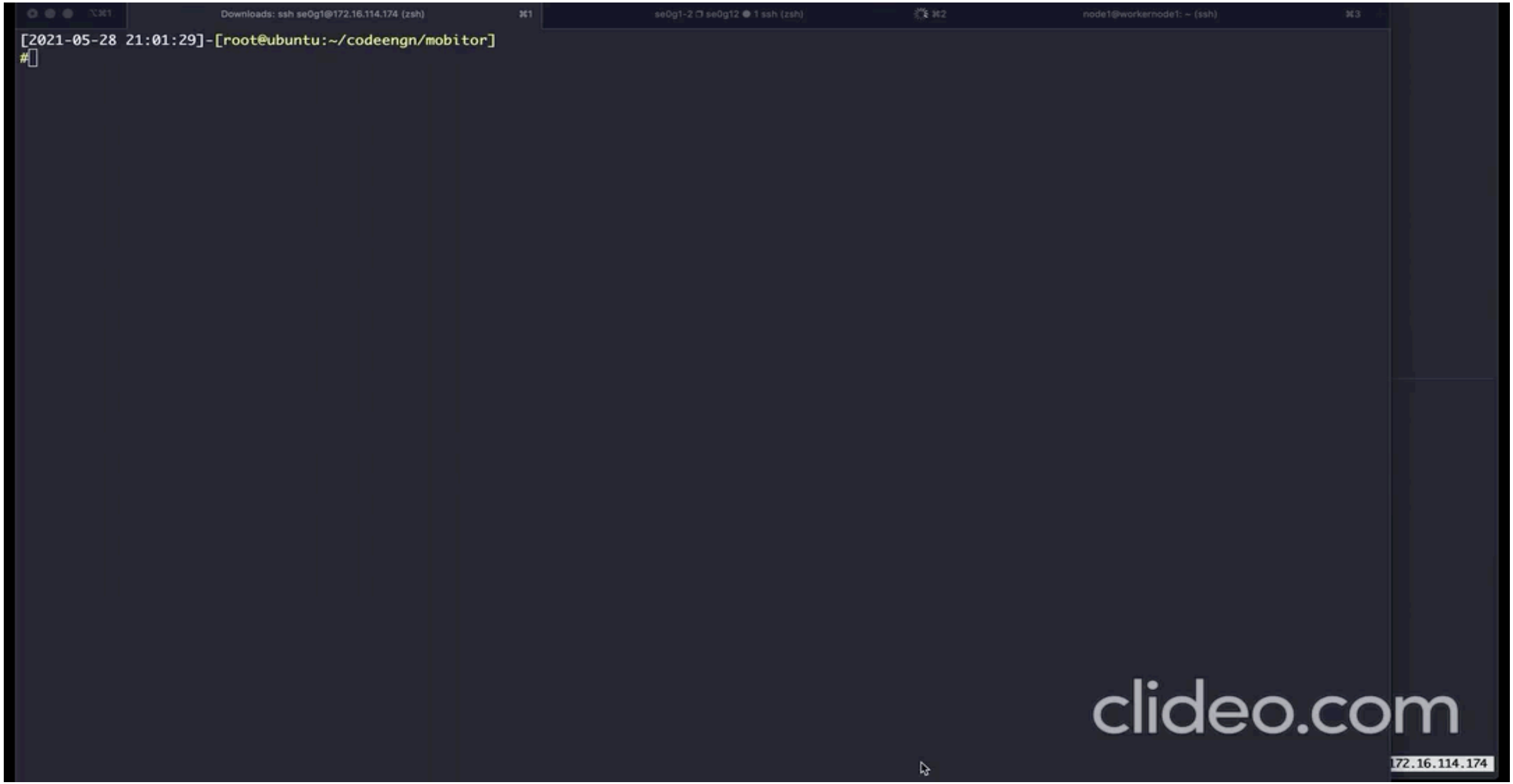
→ html 파일 저장(실시간)

- **최신 Kubernetes에서 사용하는 weave(CNI plugin)에서 위반 상황 탐지 (취약점 발견)**

컨테이너 버그 찾기 – Introspection Tool



컨테이너 버그 찾기 – Introspection Tool



컨테이너 버그 찾기 – Introspection Tool

공격 시나리오

1. 다른 POD에 명령을 내릴 수 있는 1Day / RBAC 가 존재
2. PID Namespace가 적용된 라이브러리에 실행시킬 명령어 추가
3. /proc file system에 접근하여, 다른 POD의 Host에 임의 파일 생성 및 실행 가능

영향도

1. Host로 탈출이 가능한 POD 중에 master node의 권한을 가진 pod의 경우 다른 pod들을 제어할 수 있게 됨
2. 서비스 운영 중인 사이트의 POD의 host로 escape이 될 경우 Host의 주요 자원에 접근하여 가용성 침해
3. 인증서 파일 및 설정 파일들을 탈취 가능

컨테이너 버그 찾기 – Introspection Tool

패치전

```
volumeMounts:
        - name: xtables-lock
          mountPath: /run/xtables.lock
          readOnly: false
      hostNetwork: true
      dnsPolicy: ClusterFirstWithHostNet
      hostPID: true
      restartPolicy: Always
      securityContext:
        seLinuxOptions: {}
      serviceAccountName: weave-net
```

Weave.yaml

패치후

```
volumeMounts:
        - name: xtables-lock
          mountPath: /run/xtables.lock
    dnsPolicy: ClusterFirstWithHostNet
    hostNetwork: true
    initContainers:
    - name: weave-init
    command:
      - /home/weave/init.sh
    image: 'docker.io/weaveworks/weave-kube:2.8.1'
```

Weave.yaml

Threat Modeling

CVE-2020-29389
CVE-2020-35467
CVE-2020-35468
CVE-2020-35190
…
총 36개

CodeQL For GO (Code Auditing)

오픈소스 기여
잘못된 코드 패턴 탐지

Introspection Tool

CVE-2020-26278

감사합니다.

How To Find Container Platform Bug?

Team. MobyDick