Готовим контейнеры полезно и вкусно

@rusdacent

@aleksey0xffd

whoami (Анатолий Карпенко)

- Автоматизатор автоматизации в Luntry
- Любитель митапошных форматов: SPb Reliability Meetup, ITGM, TechTrain, DevOops, DEFCON'ы, B4CKSP4CE, DevOps40
- Веду канал «Технологический Болт Генона»
- Рисую несмешные мемы

whoami (Алексей Федулаев)

- Руководитель направления Cloud Native Security в MTC Web Services
- В ИБ с 2011 года
- Специалист по безопасности
- Автор канала «Ever Secure»

tg: @aleksey0xffd

С чего начинается готовка?

С чего начинается готовка?

С выбора рецепта

В нашем случае Dockerfile

С чего начинается готовка?

Без какого ингридиента невозможен Dockerfile?

С чего начинается готовка?

FROM reci:pe

Base image

Откуда берём?

Base image

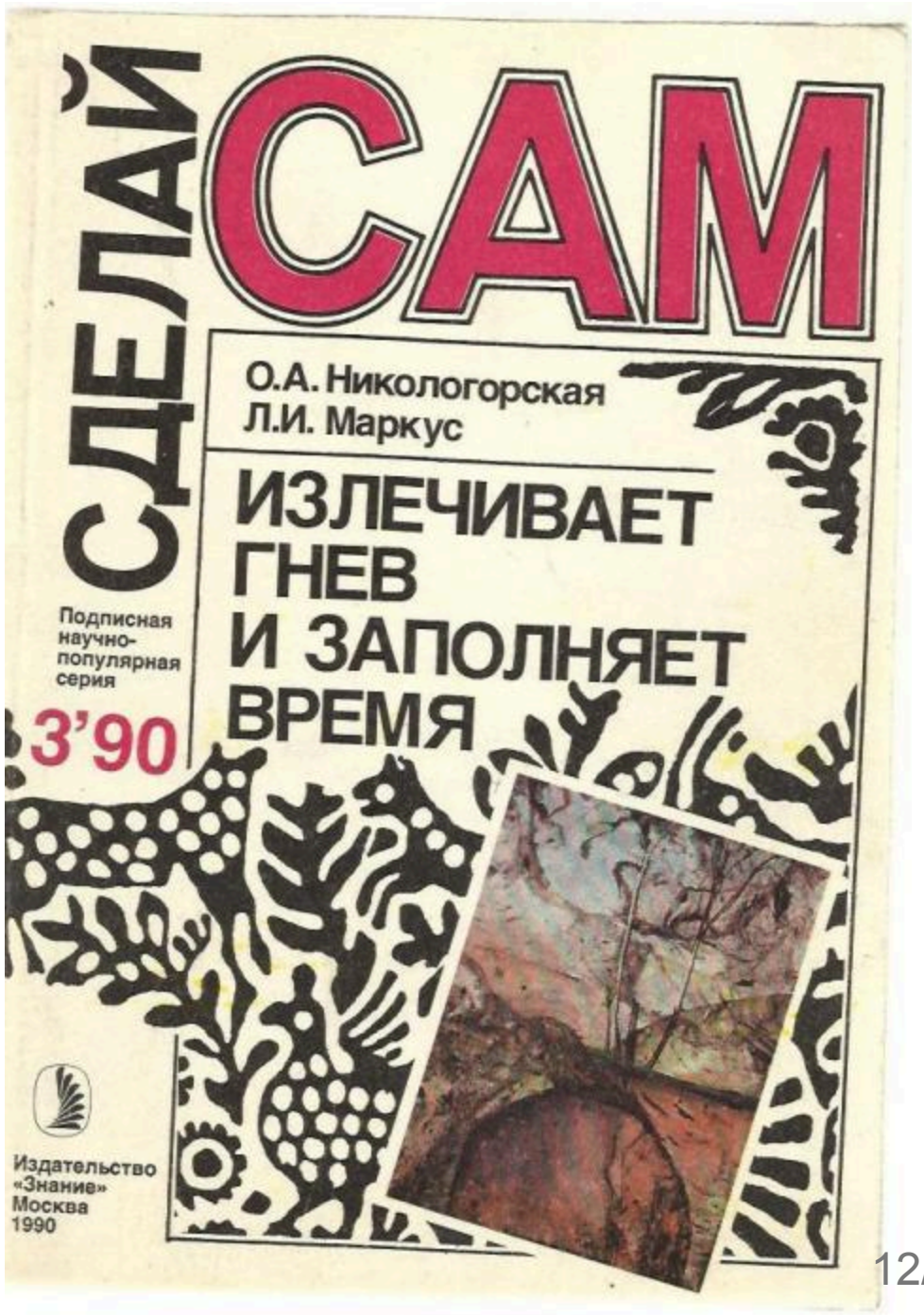
Откуда берём?

- DockerHub, ghcr.io (GitHub), quay.io (RedHat),...

Base image

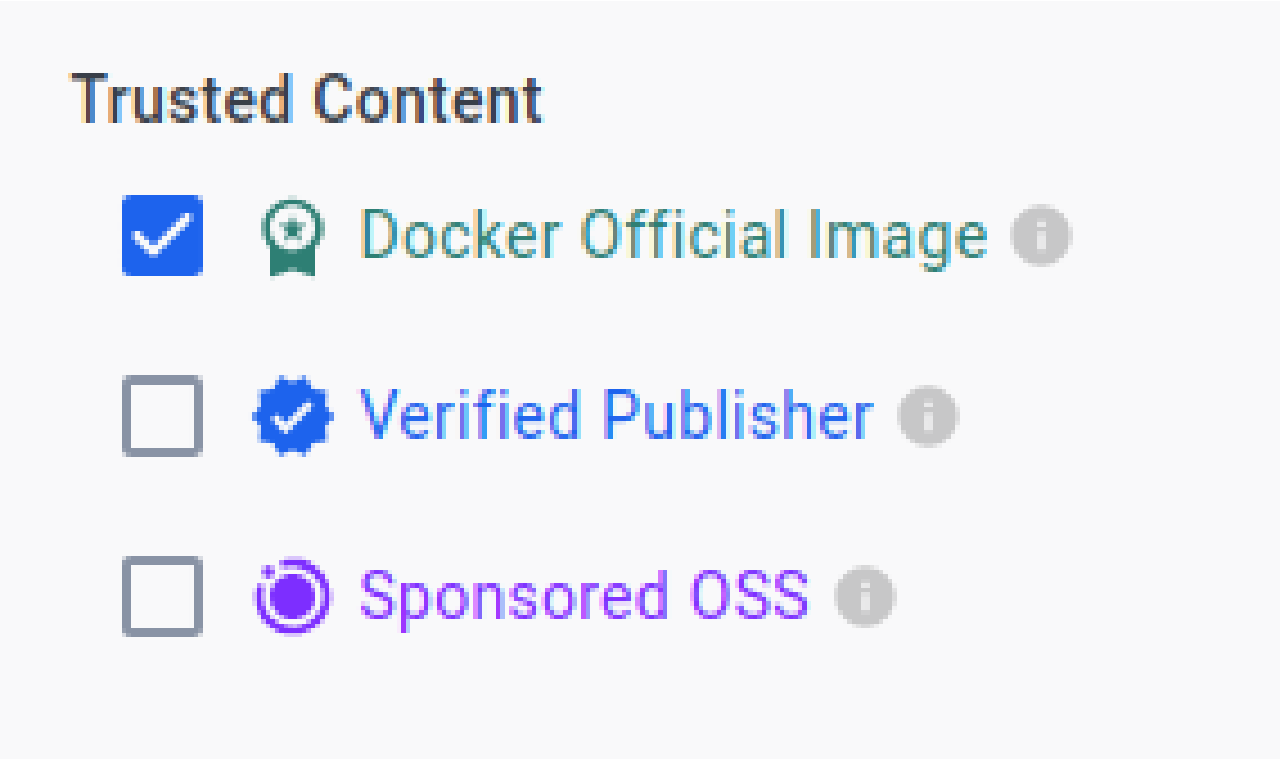
Откуда берём?

- DockerHub, ghcr.io (GitHub), quay.io (RedHat),...
- Собираем сами



DockerHub

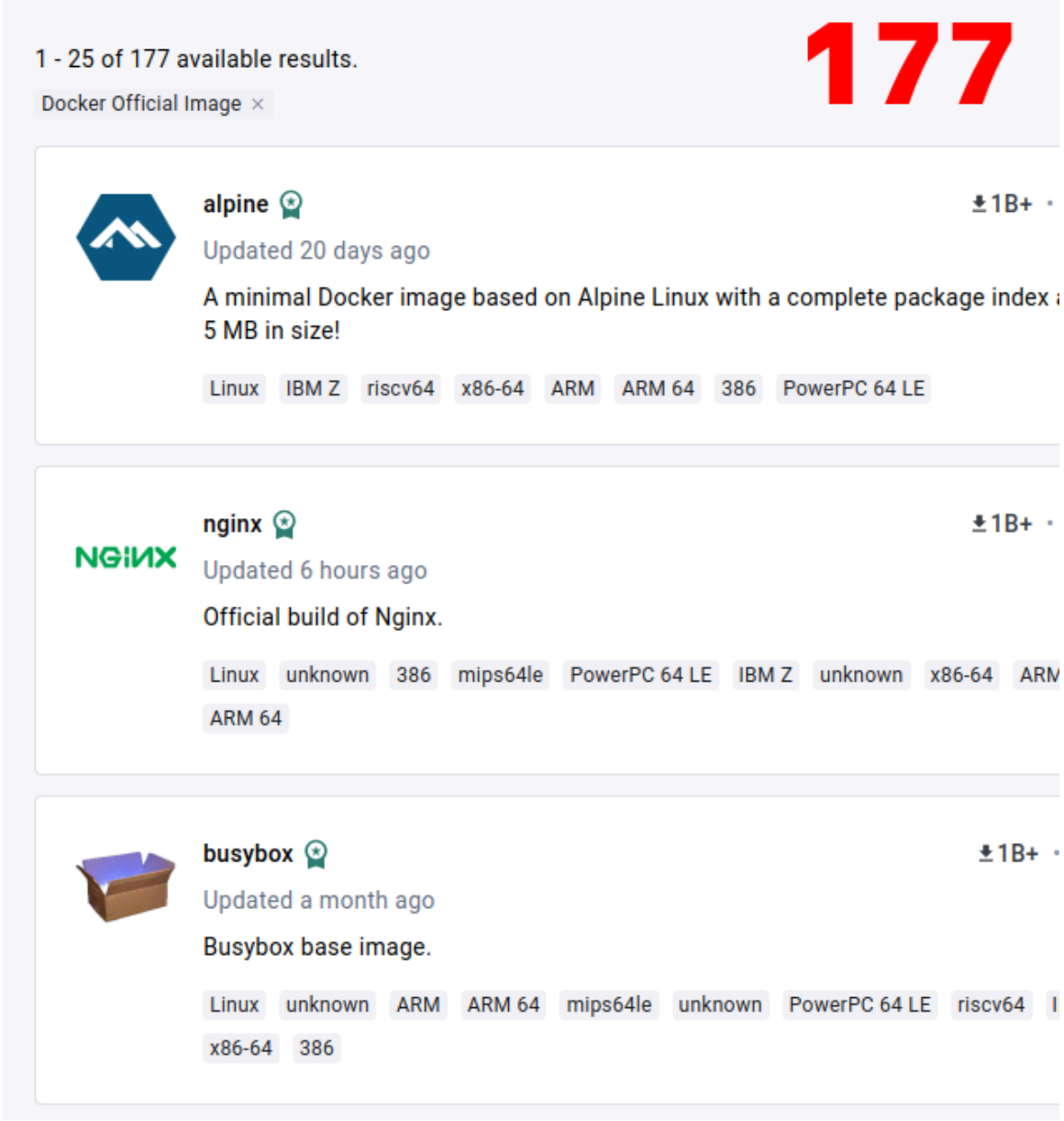
Каким доверяем?



DockerHub

Official

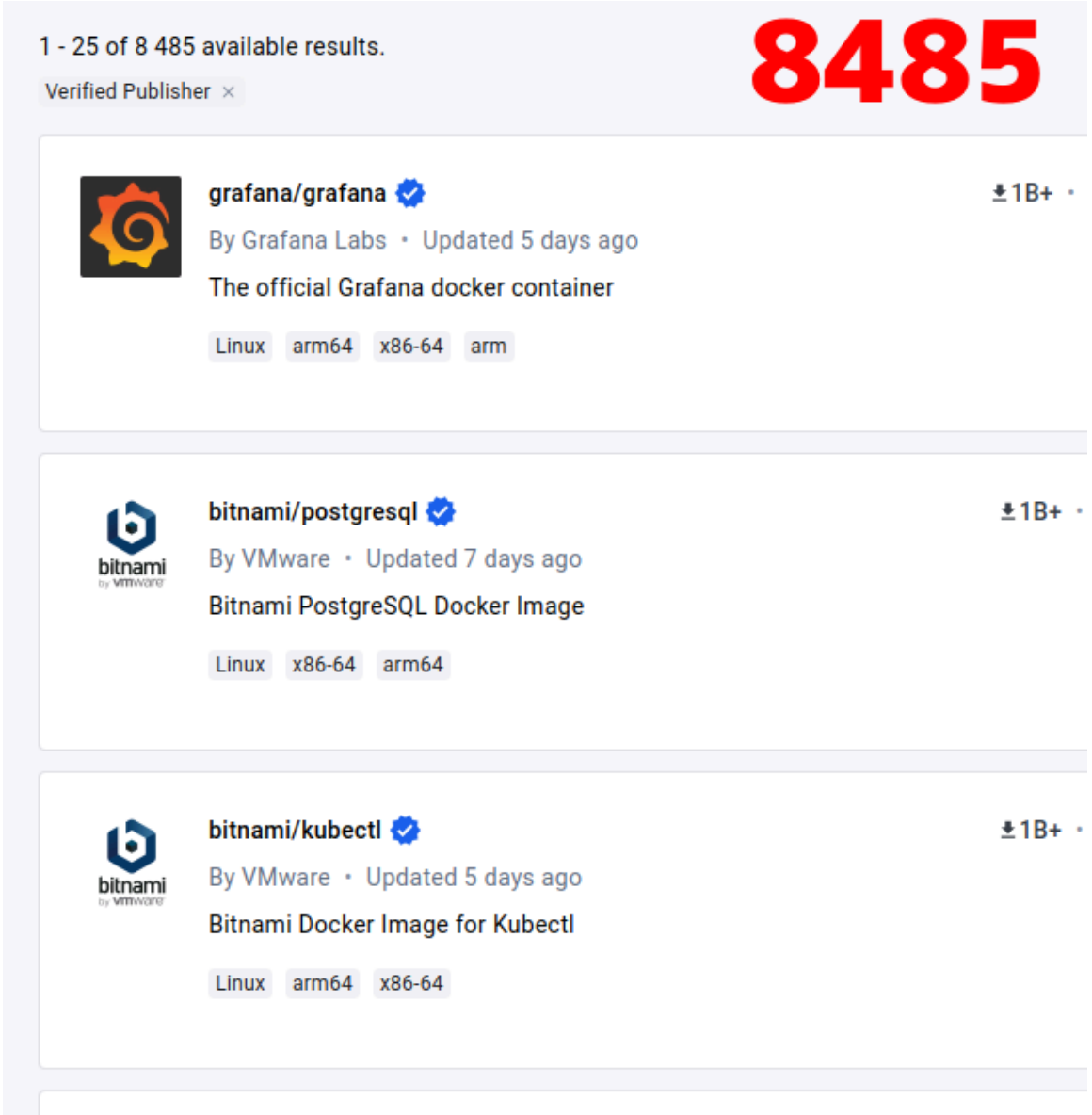
> Docker Official Images are a curated set of Docker open source and "drop-in" solution repositories.



DockerHub

Verified Publisher

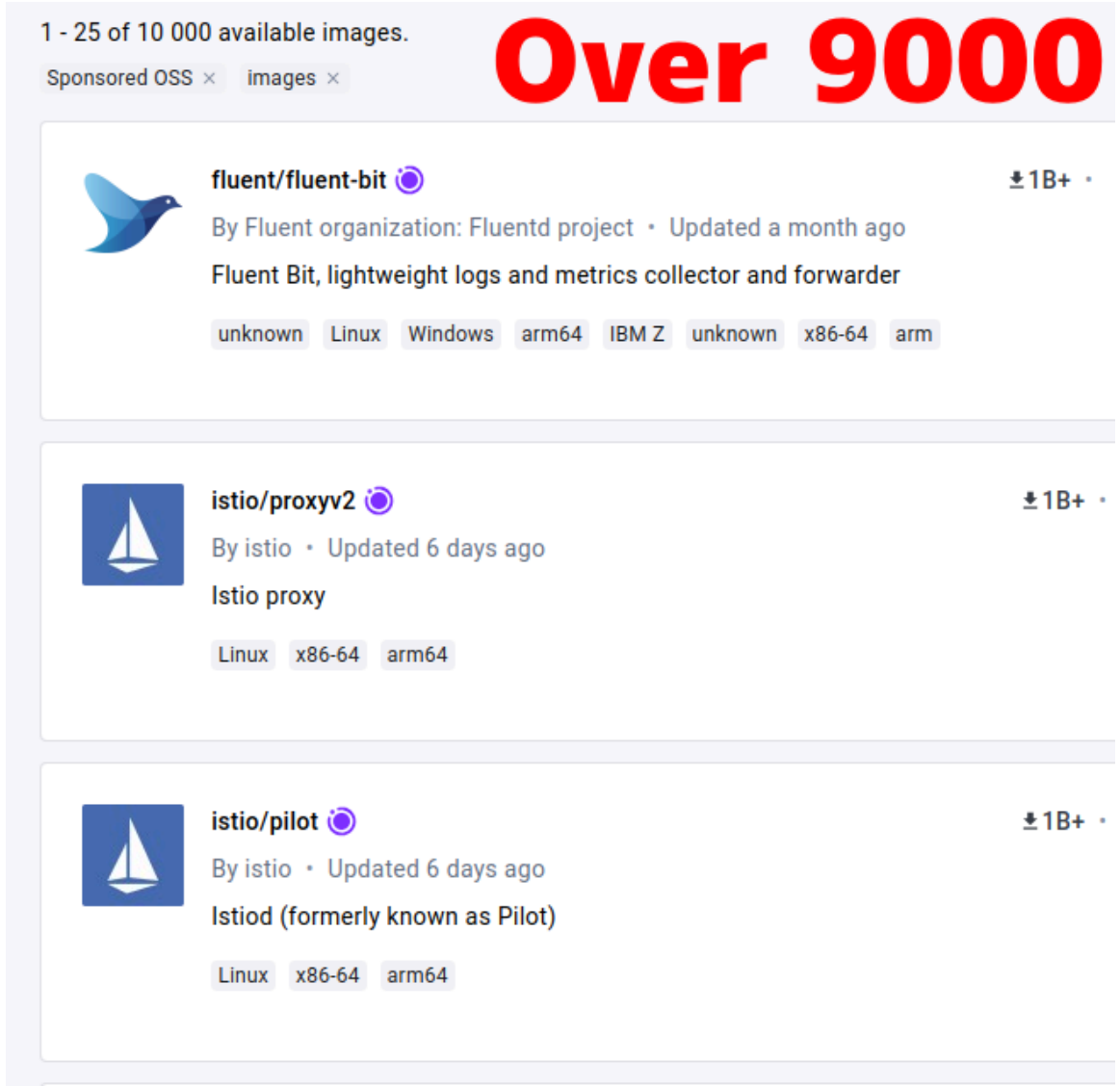
> High-quality images from publishers verified by Docker. These products are published and maintained directly by a commercial entity. These images are not subject to rate limiting.



DockerHub

Sponsored OSS

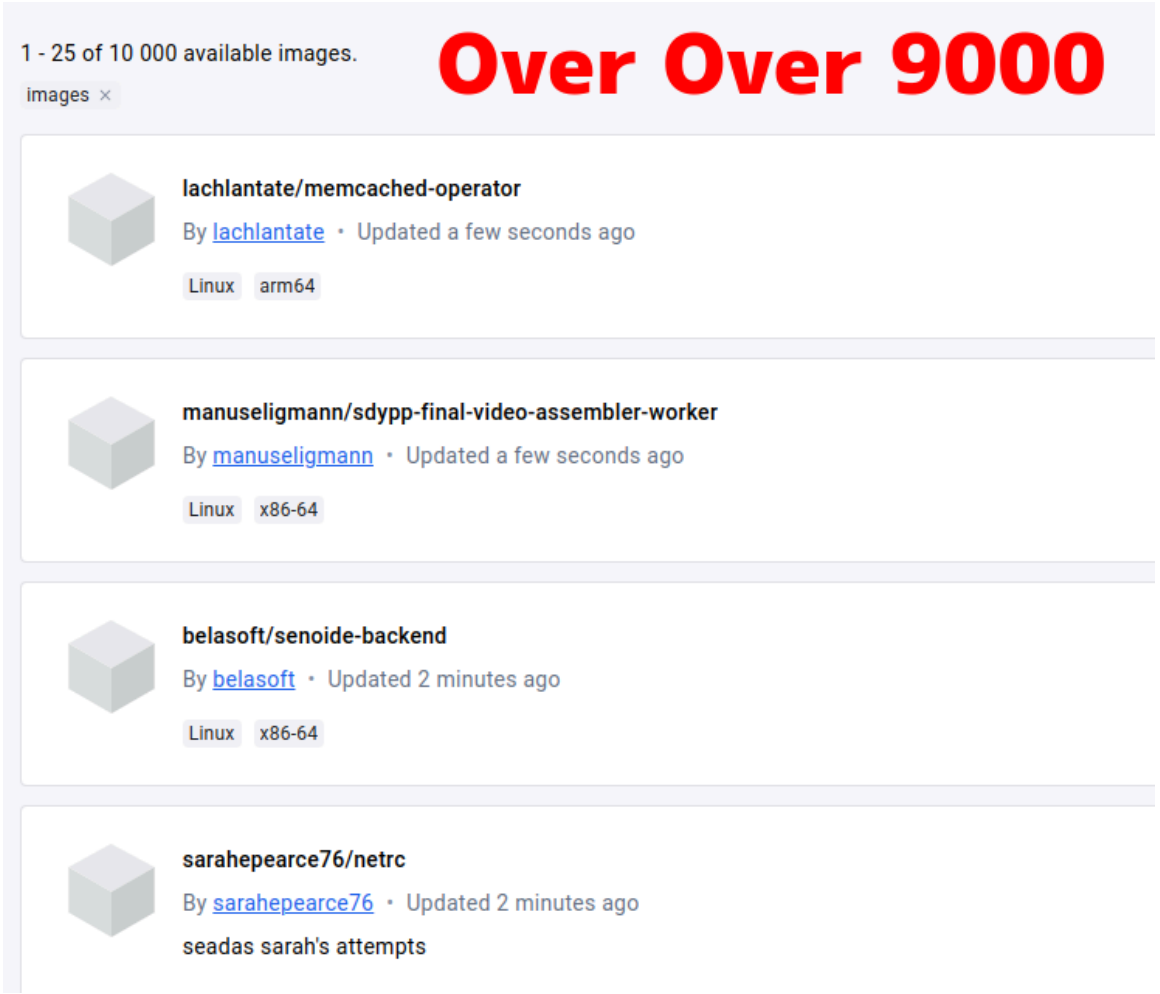
> Docker-Sponsored Open Source Software. These are images published and maintained by open-source projects that are sponsored by Docker through our open source program.



DockerHub

Other

> to the 150 million images created by the Docker community (2020)



DockerHub (Ищем образ)

Ищем на

https://hub.docker.com/

образ `kafka`

DockerHub (Ищем образ)

Ищем на
https://hub.docker.com/
образ `kafka`

Какой выберем?
Почему?

С чего начинается готовка?

С мук выбора

Base image

Кто знает что у него в базовом образе?

Base image

Кто собирал базовый образ сам?

Какие?

Зачем?

Base image (Как?)

- Из файловой системы
- Из Scratch
- Из iso

. . .

```
root@docker:~# docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
root@docker:~# echo $(openssl rand -base64 32) > myimage/secret.txt
root@docker:~# tar -C myimage/ -c . -f myimage.tar
root@docker:~# docker import myimage.tar myimage:latest
sha256:64d777b0254f87ba5277b2e9c6e1f864c05b1740480b75c2d8d0e70fdfc3173f
root@docker:~# docker images
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
myimage      latest    64d777b0254f   6 seconds ago   1.07GB
root@docker:~#
```

Base image (Ссылки)

Ссылки

- Base images (via Docker)
- Moby GitHub repo
- Creating your Own Base Image for Docker

Конкретный пример

```
$ docker run -it --rm redos-7.3:latest cat /etc/redos-release


RED OS release MUROM (7.3.2)
```

SAST для Dockerfile

SAST для Dockerfile

Кто сканит Dockerfile?

Что используется?

Hadolint

- Опирается на рекомендации Docker

Hadolint

- Опирается на рекомендации Docker
- Использует ShellCheck для скриптов в `RUN`

Hadolint

- Опирается на рекомендации Docker
- Использует ShellCheck для скриптов в `RUN`
- OpenSource

Hadolint (Плюсы)

- Опирается на рекомендации Docker
- Использует ShellCheck для скриптов в `RUN`
- OpenSource
- Расширямость

Hadolint (Тоже плюс)

- Haskell

Hadolint (DL3003)

```haskell
module Hadolint.Rule.DL3003 (rule) where

import Hadolint.Rule
import Hadolint.Shell (ParsedShell, usingProgram)
import Language.Docker.Syntax (Instruction (..), RunArgs (..))

rule :: Rule ParsedShell
rule = simpleRule code severity message check
  where
    code = "DL3003"
    severity = DLWarningC
    message = "Use WORKDIR to switch to a directory"
    check (Run (RunArgs args _)) = foldArguments (not . usingProgram "cd") args
    check _ = True
{-# INLINEABLE rule #-}
```

https://github.com/hadolint/hadolint/blob/master/src/Hadolint/Rule/DL3003.hs

Hadolint (DL3049)

```
-- (67-85)

markGood :: Acc -> Acc
markGood Empty = Empty
markGood (Acc stageid good silent bad) =
  Acc stageid (good |> Set.insert stageid) (silent |> Set.delete stageid) (bad |> Set.delete stageid)

markSilentByAlias :: Text.Text -> Acc -> Acc
markSilentByAlias _ Empty = Empty
markSilentByAlias silentname (Acc stageid good silent bad) =
  Acc stageid good (silent |> Set.union stages) (bad |> remove stages)
  where
    stages = Set.filter byName bad
    byName (StageID BaseImage {alias = Nothing} _) = False
    byName (StageID BaseImage {alias = Just als} _) = unImageAlias als == silentname
    remove set fromThis = Set.difference fromThis set

getCurrentStageName :: State Acc -> Text.Text
getCurrentStageName (State _ (Acc (StageID BaseImage {image, alias = Nothing} _) _ _ _)) = imageName image
getCurrentStageName (State _ (Acc (StageID BaseImage {alias = Just als} _) _ _ _)) = unImageAlias als
getCurrentStageName _ = ""
```

https://github.com/hadolint/hadolint/blob/master/src/Hadolint/Rule/DL3049.hs

DL3013

Vlastimil Zeman edited this page on Nov 1, 2017 · 3 revisions

Pin versions in pip.

Problematic code:

```
FROM python:3.4
RUN pip install django
RUN pip install https://github.com/Banno/carbon/tarball/0.9.x-fix-events-callback
```

Correct code:

```
FROM python:3.4
RUN pip install django==1.9
RUN pip install git+https://github.com/Banno/carbon@0.9.15
```

https://habr.com/ru/companies/flant/articles/787494/

Hadolint (Ищем проблемы)

Demo

https://github.com/rusdacent/cooking-docker-containers/tree/main/hadolint

Hadolint (Обсуждаем проблемы - DL3020)

> Use COPY instead of ADD for files and folders

```
ADD demo /home/myapp/demo
```

> ADD instruction introduces risks such as adding malicious files from URLs without scanning and unpacking procedure vulnerabilities.

Bonus

```
ADD --checksum=sha256:24454f830cdb571e2c4ad15481119c43b3cafd48dd869a9b2945d1036d1dc68d \
      https://mirrors.edge.kernel.org/pub/linux/kernel/Historic/linux-0.01.tar.gz /
```

https://github.com/hadolint/hadolint/wiki/DL3020

Hadolint (online)

```
1 FROM scratch as base
  <string>:2:18:
  |
  2 | COPY --chmod=777 hadolint /bin/hadolint
  | ^
  invalid flag: --chmod
2 COPY --chmod=777 hadolint /bin/hadolint
3 LABEL org.opencontainers.image.source="https://github.com/hado
4 CMD ["/bin/hadolint", "-"]
5
6 FROM debian:bullseye-slim as debian
7 COPY --chmod=777 hadolint /bin/hadolint
8 LABEL org.opencontainers.image.source="https://github.com/hado
9
```

https://hadolint.github.io/hadolint/

Hadolint (COPY)

COPY

COPY has two forms:

```
COPY [--chown=<user>:<group>] [--chmod=<perms>] <src>... <dest>
COPY [--chown=<user>:<group>] [--chmod=<perms>] ["<src>",... "<dest>"]
```

This latter form is required for paths containing whitespace

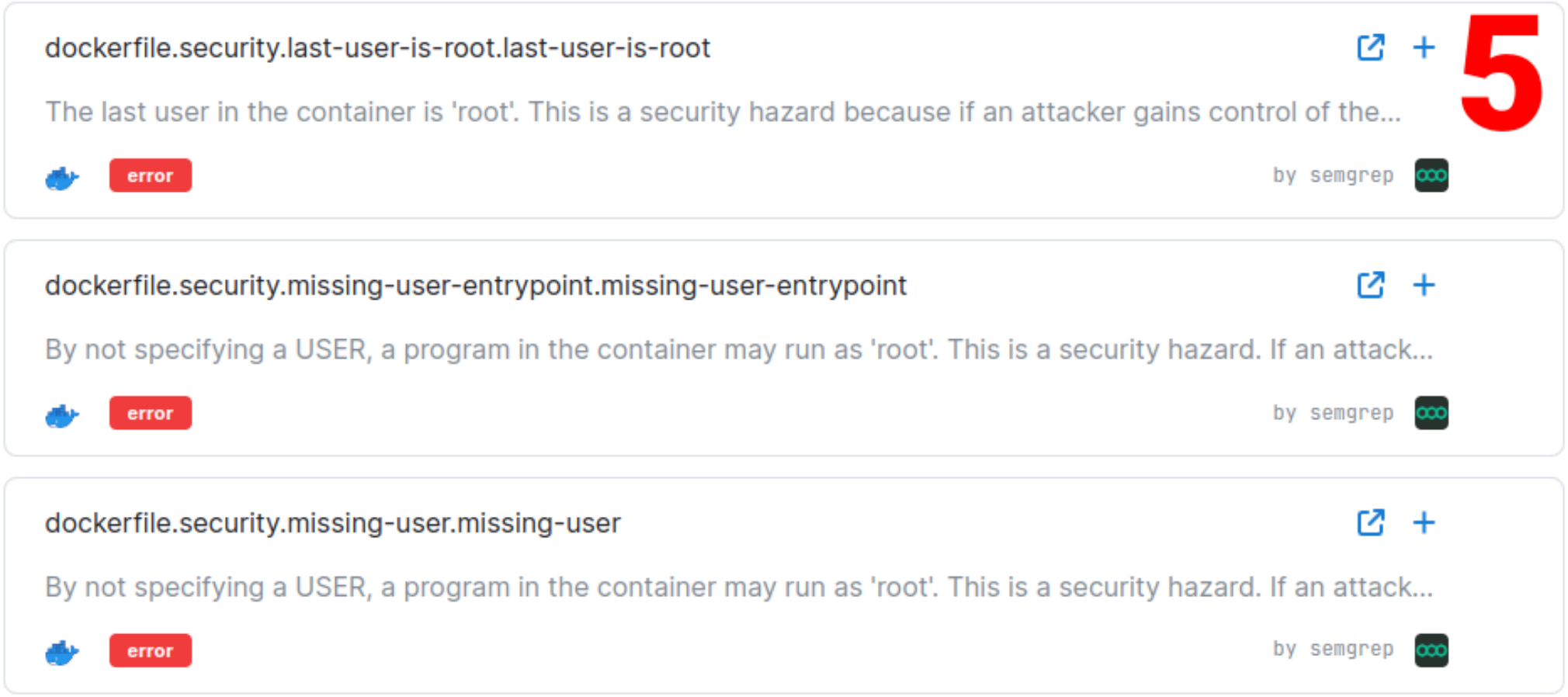
https://docs.docker.com/reference/dockerfile/#copy---chown---chmod

Semgrep

Semgrep

Кто использует Semgrep?

Semgrep (ruleset)



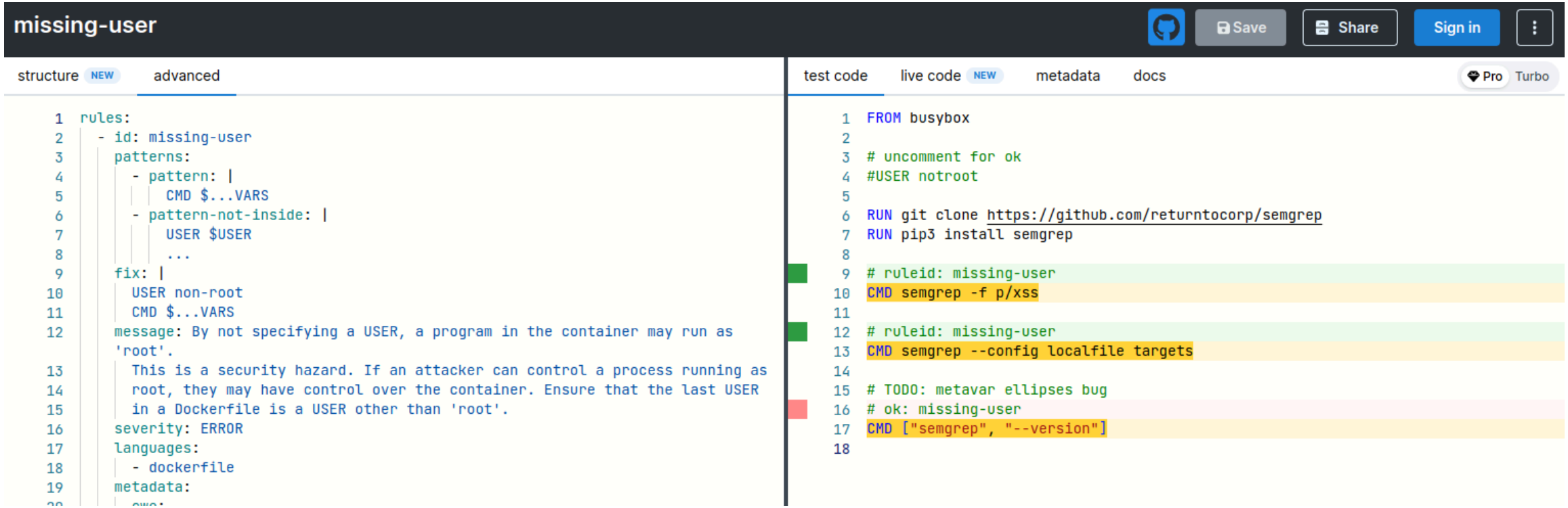
https://semgrep.dev/p/dockerfile

Semgrep (GitHub rules)

```
$ tree -P '*.yaml' dockerfile/
dockerfile/
├── audit
│   └── dockerfile-source-not-pinned.yaml
├── best-practice
│   ├── avoid-apk-upgrade.yaml
. . .
│   └── use-workdir.yaml
├── correctness
│   ├── invalid-port.yaml
│   ├── missing-assume-yes-switch.yaml
│   └── multiple-entrypoint-instructions.yaml
└── security
    ├── dockerd-socket-mount.yaml
    ├── last-user-is-root.yaml
    ├── missing-user-entrypoint.yaml
    ├── missing-user.yaml
    ├── no-sudo-in-dockerfile.yaml
    └── secret-in-build-arg.yaml

4 directories, 36 files
```

https://github.com/semgrep/semgrep-rules

Semgrep (playground)



https://semgrep.dev/playground/r/dockerfile.security.missing-user-entrypoint.missing-user-entrypoint?editorMode=advanced

Semgrep (Ищем проблемы)

Demo

Checkov

Checkov

> Scans Terraform, Terraform Plan, Terraform JSON, CloudFormation, AWS SAM, Kubernetes, Helm, Kustomize, Dockerfile, Serverless framework, Ansible, Bicep, ARM, and OpenTofu template files.

> Scans Argo Workflows, Azure Pipelines, BitBucket Pipelines, Circle CI Pipelines, GitHub Actions and GitLab CI workflow files

https://github.com/bridgecrewio/checkov

Checkov (example)

```
FROM node:alpine
WORKDIR /usr/src/app
COPY package*.json ./
RUN npm install
COPY . .
EXPOSE 3000 22
HEALTHCHECK CMD curl --fail http://localhost:3000 || exit 1
CMD ["node","app.js"]
```

```
$ checkov -d . --framework dockerfile
```

https://www.checkov.io/7.Scan Examples/Dockerfile.html

Checkov (Ищем проблемы)

Demo

Checkov (Rules, Python)

```python
class RootUser(BaseDockerfileCheck):
    def __init__(self) -> None:
        name = "Ensure the last USER is not root"
        id = "CKV_DOCKER_8"
        supported_instructions = ("USER",)
        categories = (CheckCategories.APPLICATION_SECURITY,)
        super().__init__(name=name, id=id, categories=categories, supported_instructions=supported_instructions)

    def scan_resource_conf(self, conf: list[_Instruction]) -> tuple[CheckResult, list[_Instruction] | None]:
        last_user = conf[-1]
        if last_user["value"] == "root":
            return CheckResult.FAILED, [last_user]

        return CheckResult.PASSED, [last_user]

```

https://github.com/bridgecrewio/checkov/blob/main/checkov/dockerfile/checks/RootUser.py

Checkov (Rules, YAML easy)

```
metadata:
  id: "CKV2_DOCKER_2"
  name: "Ensure that certificate validation isn't disabled with curl"
  category: "APPLICATION_SECURITY"
definition:
    cond_type: attribute
    resource_types:
      - RUN
    attribute: value
    operator: not_regex_match
    value: ".*(curl[^\\|&;]*\\s+((--insecure)|(-[^-\\s]*k))).*"
```

https://github.com/bridgecrewio/checkov/blob/main/checkov/dockerfile/checks/graph_checks/RunUnsafeCurl.yaml

Checkov (Rules, YAML hard)

```yaml
metadata:
  id: "CKV2_DOCKER_12"
  name: "Ensure that certificate validation isn't disabled for npm via the 'NPM_CONFIG_STRICT_SSL' environmnet variable"
  category: "APPLICATION_SECURITY"
definition:
  or:
    - cond_type: attribute
      resource_types:
        - ARG
        - ENV
      attribute: value
      operator: not_regex_match
      value: "(.*\\s+)?(((NPM_CONFIG_STRICT_SSL)|(npm_config_strict_ssl))(=|\\s+)((false)|('false')|(\"false\"))).*"
    - cond_type: attribute
      resource_types:
        - RUN
      attribute: value
      operator: not_regex_match
      value: "(.*[\\s;&|]+)?(((NPM_CONFIG_STRICT_SSL)|(npm_config_strict_ssl))=((false)|('false')|(\"false\"))) .*"
```

https://github.com/bridgecrewio/checkov/blob/main/checkov/dockerfile/checks/graph_checks/EnvNpmConfigStrictSsl.yaml

SAST для Dockerfile

Что ещё может быть?

Что бы вы хотели ловить в своих Dockerfile?

SAST для Dockerfile

Всё что угодно

Leaky Vessels (Описание)

> CVE-2024-21626: уязвимость типа order-of-operations в команде WORKDIR в runc. Позволяет получить несанкционированный доступ к операционной системе хоста и потенциально скомпрометировать всю систему.

> CVE-2024-23651: состояние гонки в Buildkit, приводящее к непредсказуемому поведению

> CVE-2024-23652: проблема, позволяющая удалять произвольные файлы или каталоги на этапе удаления контейнера в Buildkit

> CVE-2024-23653: уязвимость возникает из-за недостаточной проверки привилегий в интерфейсе GRPC Buildkit

https://xakep.ru/2024/02/05/leaky-vessels/

Leaky Vessels (Пример)

```
$ runc --version
runc version 1.1.7-0ubuntu1~22.04.2
. . .
```

```
cat /proc/11/cwd/../../../../etc/passwd
root: x:0:0:root:/root:/bin/bash
messagebus: x:498:498:User for D-Bus:/run/dbus:/usr/sbin/nologin
lp: x:497:497:Printing daemon:/var/spool/lpd:/usr/sbin/nologin
systemd-timesync: x:484:484:systemd Time
. . .
```

https://nitroc.org/en/posts/cve-2024-21626-illustrated/

Leaky Vessels (Правила)

```
common.Rule{
  ID: 1,
  CVE: "CVE-2024-21626",
  Name: "runc process.cwd & Leaked fds Container Breakout",
  Blogpost: "https://snyk.io/blog/cve-2024-21626-runc-process-cwd-container-breakout",
  Inst:        "WORKDIR",
  ArgRegex:    regexp2.MustCompile("/proc/self/fd/", 0),
},
common.Rule{
  ID: 2,
  CVE: "CVE-2024-23651",
  Name: "Buildkit Mount Cache Race: Build-time Race Condition Container Breakout",
  Blogpost: "https://snyk.io/blog/cve-2024-23651-docker-buildkit-mount-cache-race",
  Inst:        "RUN",
  ArgRegex:    regexp2.MustCompile("--mount=type=cache", 0),
},
common.Rule{
. . .
```

https://github.com/snyk/leaky-vessels-static-detector/

Leaky Vessels (repo)

Умеет не только Dockerfile, но и образы

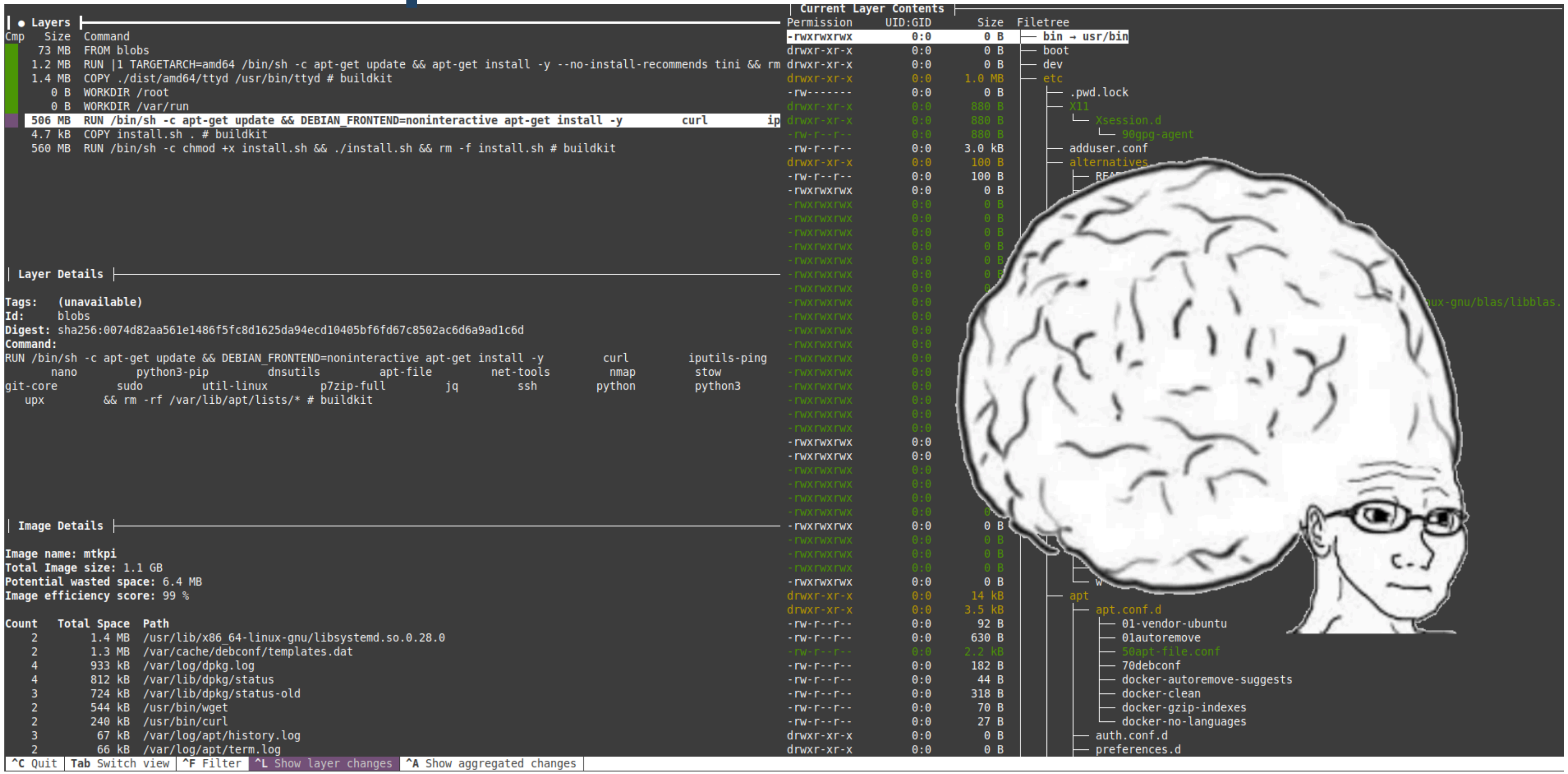
- Regex based detection rule matching.
- Dockerfile detection
- Image detection

https://github.com/snyk/leaky-vessels-static-detector/

Что на счёт образов?

https://youtube.com/watch?v=9IGXZStnhS8

Что на счёт образов?



Dive

Demo

Что на счёт образов?

Есть что!

Dockle

Проверяет образ на

- CIS Docker Benchmark <- Центр интернет-безопасности (CIS)
- best-practice for Dockerfile

https://github.com/goodwithtech/dockle

Dockle

Demo

Dockle (CIS-DI-0010 - Files)

```
func (a CredentialAssessor) RequiredFiles() []string {
        return []string{
                "credentials.json",
                "credential.json",
                // TODO: Only check .docker/config.json
                // "config.json",
                "credentials",
                "credential",
                "password.txt",
                "id_rsa",
                "id_dsa",
                "id_ecdsa",
                "id_ed25519",
                "secret_token.rb",
                "carrierwave.rb",
                "omniauth.rb",
                "settings.py",
                "database.yml",
                "credentials.xml",
        }
}
```

https://github.com/goodwithtech/dockle/blob/master/pkg/assessor/credential/credential.go

Dockle (CIS-DI-0010 - Extensions)

```go
func (a CredentialAssessor) RequiredExtensions() []string {
        return []string{
                // reference: https://github.com/eth0izzle/shhgit/blob/master/config.yaml
                // TODO: potential sensitive data but they have many false-positives.
                //       Dockle need to analyze each file.
                //".pem",
                //".key",
                //".p12",
                //".pkcs12",
                //".pfx",
                //".asc",
                ".secret",
                ".ovpn",
                ".private_key",
                ".cscfg",
                ".rdp",
                ".mdf",
                ".sdf",
                ".bek",
                ".tpm",
                ".fve",
                ".jks",
                ".psafe3",
                ".agilekeychain",
                ".keychain",
                ".pcap",
                ".gnucache",
        }
}
```

https://github.com/goodwithtech/dockle/blob/master/pkg/assessor/credential/credential.go

Dockle (MTKPI)

MTKPI – Multi Tool Kubernetes Pentest Image

> $ dockle r0binak/mtkpi:v1.4

> FATAL - DKL-DI-0001: Avoid sudo command
> * Avoid sudo in container : RUN /bin/sh -c apt-get update && DEBIAN_FRONTEND=noninteractive apt-get install -y curl iputils-ping nano python3-pip dnsutils apt-file net-tools nmap stow git-core sudo util-linux p7zip-full jq ssh python python3 upx && rm -rf /var/lib/apt/lists/* # buildkit
> ...

https://github.com/r0binak/MTKPI

Dockle (MTKPI, sudo)

```
RUN apt-get update \
    && DEBIAN_FRONTEND=noninteractive \
    apt-get install -y \
        curl \
. . .
        nmap \
        stow \
        git-core \
        sudo \
. . .
        python3 \
        upx \
        && rm -rf /var/lib/apt/lists/*
```

https://github.com/r0binak/MTKPI/blob/main/Dockerfile

Clair

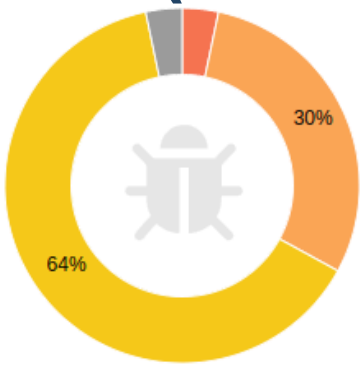
> Vulnerability Static Analysis for Containers

- RedHat
- Интеграция с Quay
- Использует отдельный пакет - Claircore

- https://secdb.alpinelinux.org/
- http://repo.us-west-2.amazonaws.com/2018.03/updates/x86_64/mirror.list
- https://cdn.amazonlinux.com/2/core/latest/x86_64/mirror.list
- https://cdn.amazonlinux.com/al2023/core/mirrors/latest/x86_64/mirror.list
- https://deb.debian.org/
- https://security-tracker.debian.org/tracker/data/json
- https://nvd.nist.gov/feeds/json/cve/1.1/
- https://linux.oracle.com/security/oval/com.oracle.elsa-*.xml.bz2
- https://packages.vmware.com/photon/photon_oval_definitions/
- https://access.redhat.com/security/data/metrics/cvemap.xml
- https://access.redhat.com/security/cve/
- https://access.redhat.com/security/data/oval/v2/PULP_MANIFEST
- https://support.novell.com/security/oval/
- https://api.launchpad.net/1.0/
- https://security-metadata.canonical.com/oval/com.ubuntu.*.cve.oval.xml
- https://osv-vulnerabilities.storage.googleapis.com/

https://github.com/quay/clair

Clair (demo)



Quay Security Scanner has detected **91** vulnerabilities.

Patches are available for **9** vulnerabilities.

- **3** High-level vulnerabilities.
- **27** Medium-level vulnerabilities.
- **58** Low-level vulnerabilities.
- **3** Unknown-level vulnerabilities.

Advisories

ADVISORY	SEVERITY	PACKAGE	CURRENT VERSION	FIXED IN VERSION	INTRODUCED IN LAYER
CVE-2024-48957	High	libarchive	3.5.3-4.el9	(None)	RUN mv -fZ /tmp/ubi.repo /etc/yum.repos.d/ub…
CVE-2024-48958	High	libarchive	3.5.3-4.el9	(None)	RUN mv -fZ /tmp/ubi.repo /etc/yum.repos.d/ub…
GHSA-78wr-2p64-hpwj	High	commons-io:commons-io	2.11.0	2.14.0	COPY tmp/cc/ /opt/cruise-control/libs/ # buildkit
CVE-2023-36632	7.5 / 10	python3	3.9.18-3.el9_…	(None)	RUN |4 JAVA_VERSION=17 TARGETOS=linux TARGETARCH=amd64 strir

https://quay.io/repository/strimzi/kafka/manifest/sha256:b29011dc978c0e3e8b445f4dc1acba97ef6a54c856047baa6422cafa0e5515fe

Trivy

Targets

- Container Image
- Filesystem
- Git Repository (remote)
- Virtual Machine Image
- Kubernetes
- AWS

Scanners

- OS packages and software dependencies in use (SBOM)
- Known vulnerabilities (CVEs)
- IaC issues and misconfigurations
- Sensitive information and secrets
- Software licenses

https://aquasecurity.github.io/trivy/

Trivy

Demo

Trivy (Allow rules)

```
{
    ID:          "tests",
    Description: "Avoid test files and paths",
    Path:        MustCompile(`(^test|\/test|-test|_test|\.test)`),
},
{
    ID:          "examples",
    Description: "Avoid example files and paths",
    Path:        MustCompile(`example`),
    Regex:       MustCompile("(?i)example"),
},
```

https://github.com/aquasecurity/trivy/blob/main/pkg/fanal/secret/builtin-allow-rules.go

Grype

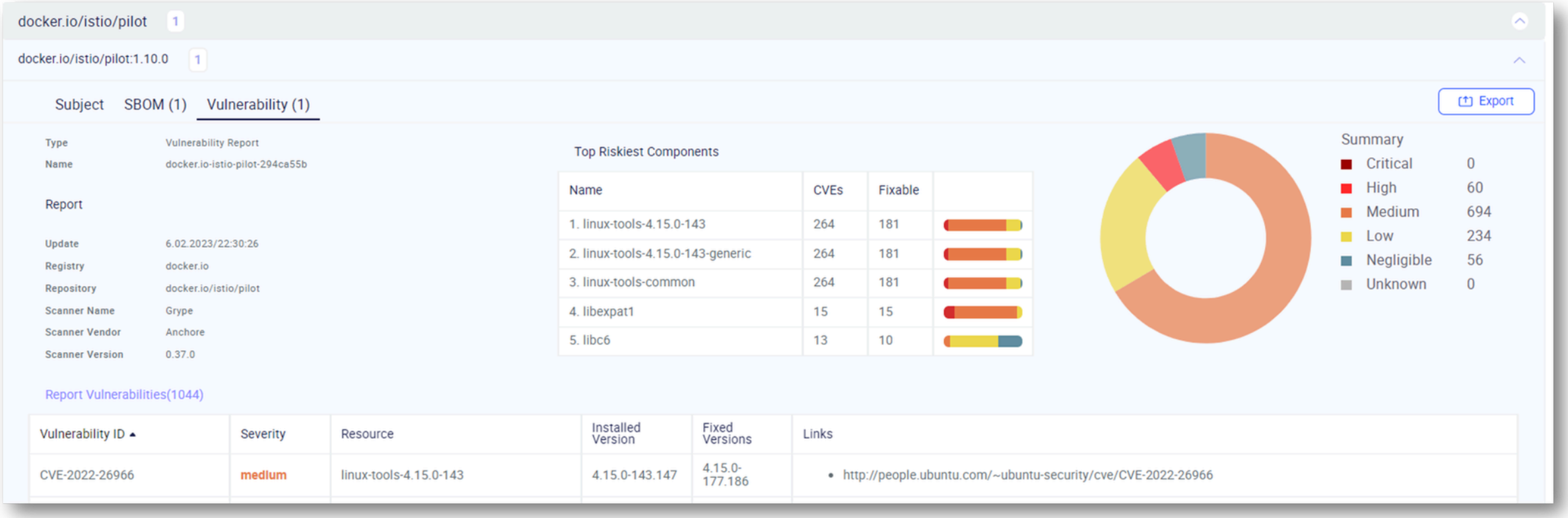
Grype + Syft ≈ Trivy

> Syft - CLI tool and library for generating a Software Bill of Materials from container images and filesystems

- Alpine Linux SecDB: https://secdb.alpinelinux.org/
- Amazon Linux ALAS: https://alas.aws.amazon.com/AL2/alas.rss
- RedHat RHSAs: https://www.redhat.com/security/data/oval/
- Debian Linux CVE Tracker: https://security-tracker.debian.org/tracker/data/json
- Github GHSAs: https://github.com/advisories
- National Vulnerability Database (NVD): https://nvd.nist.gov/vuln/data-feeds
- Oracle Linux OVAL: https://linux.oracle.com/security/oval/
- RedHat Linux Security Data: https://access.redhat.com/hydra/rest/securitydata/
- Suse Linux OVAL: https://ftp.suse.com/pub/projects/security/oval/
- Ubuntu Linux Security: https://people.canonical.com/~ubuntu-security/

https://github.com/anchore/grype

Как это выглядит в проде (Luntry)



Что дальше?

НАСКАНИРОВАЛСЯ И СПИТ

Status	Pipeline
failed	#319049365 error
failed	#317510274 error
failed	#317485818 error

Наводим порядок!

Registry

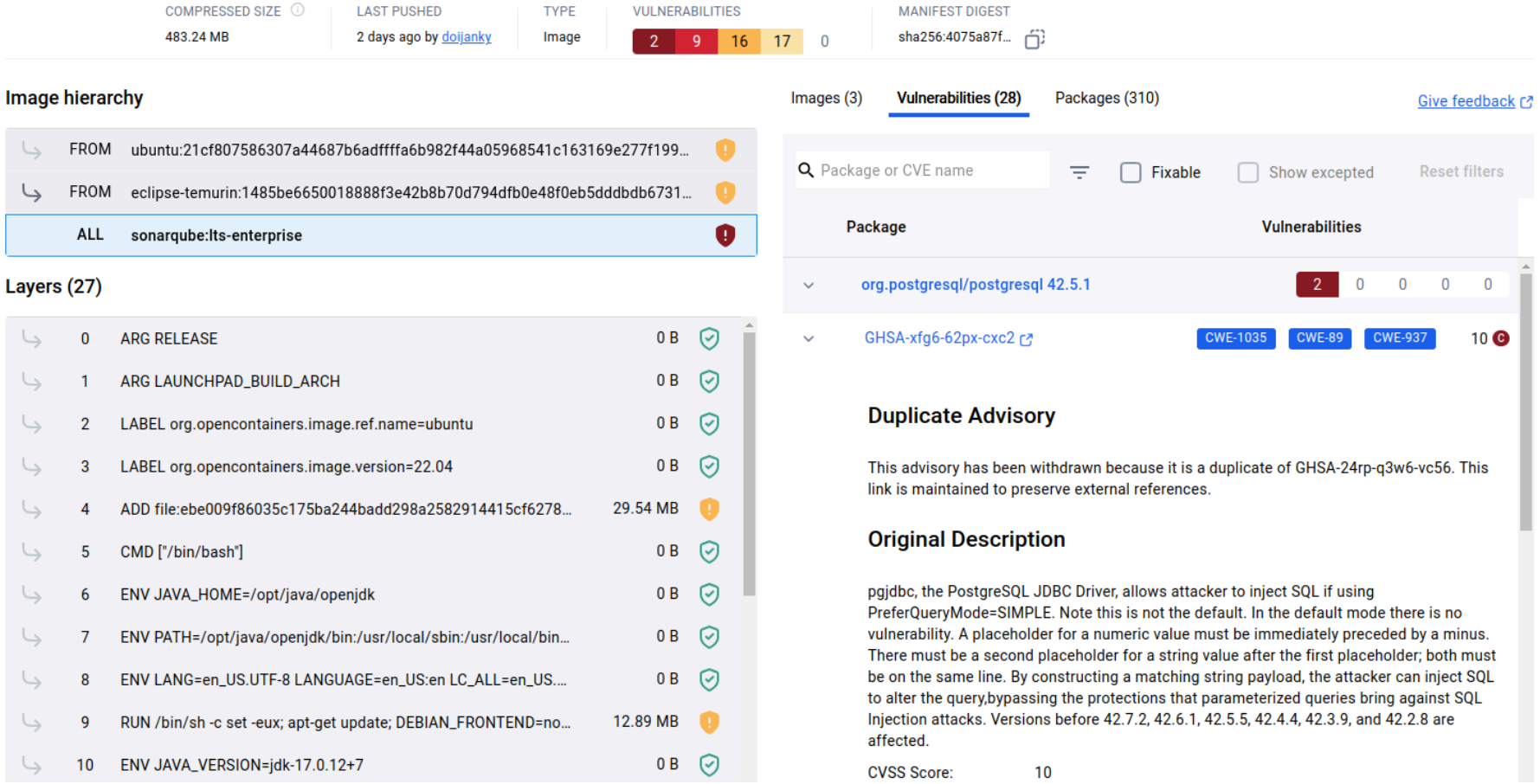
"Чужие"

- Docker Hub
- GitHub
- GitLab
- Quay
- Chainguard
- . . .

"Свои"

- registry
- GitLab
- Quay
- Harbor
- Artifactory / JFrog Container Registry
- Nexus
- . . .

Docker Hub (demo)



https://hub.docker.com/layers/library/sonarqube/lts-enterprise/images/sha256-4075a87f5512b14b67f8a480aa8132131f705b2b999001e9f28a9b3f9ec950b4?context=explore

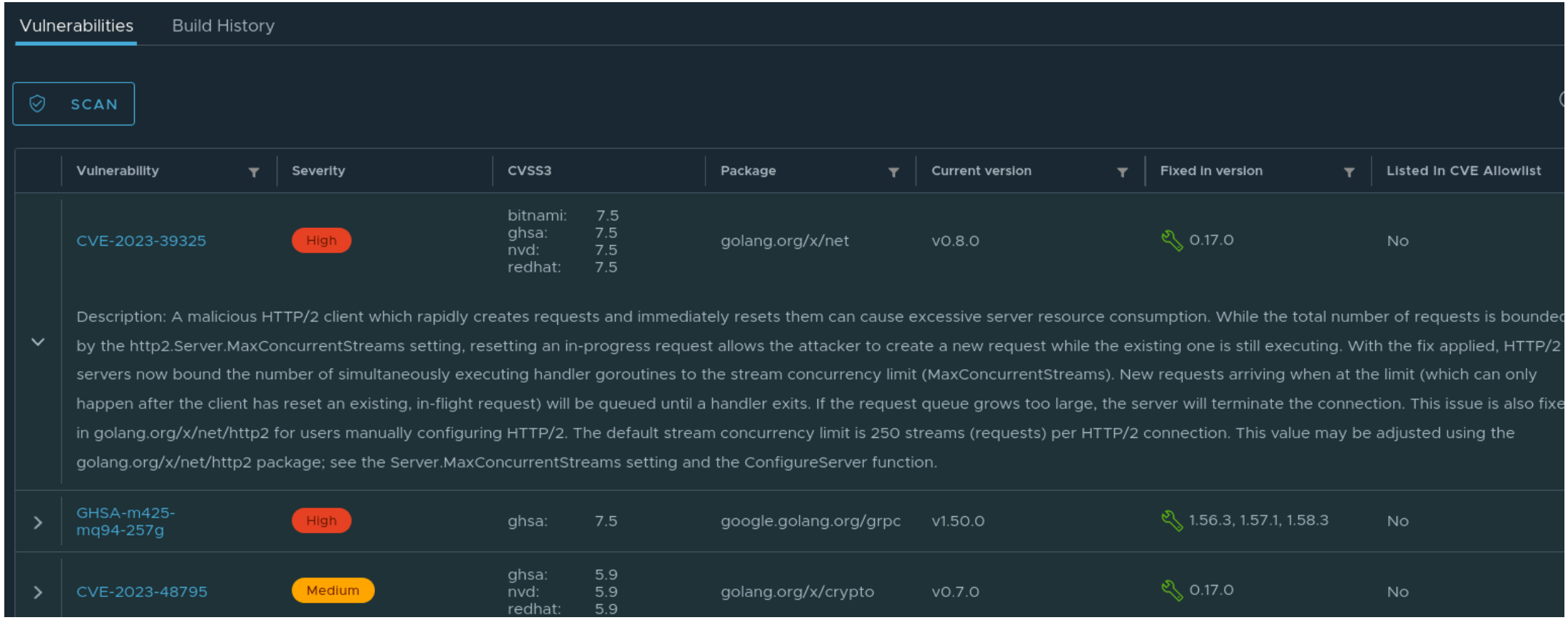
Docker Hub (Log4Shell)

Docker Hub (Docker Scout)

- Умеет в интеграцию с
  - Artifactory
  - Amazon Elastic Container Registry
  - Azure Container Registry
- Требуется платный аккаунт
- Утверждается, что доступен для sponsored и verified аккаунтов, но я не нашёл ни одного живого примера

https://docs.docker.com/scout/integrations/

Harbor (Trivy, ~~Clair~~)

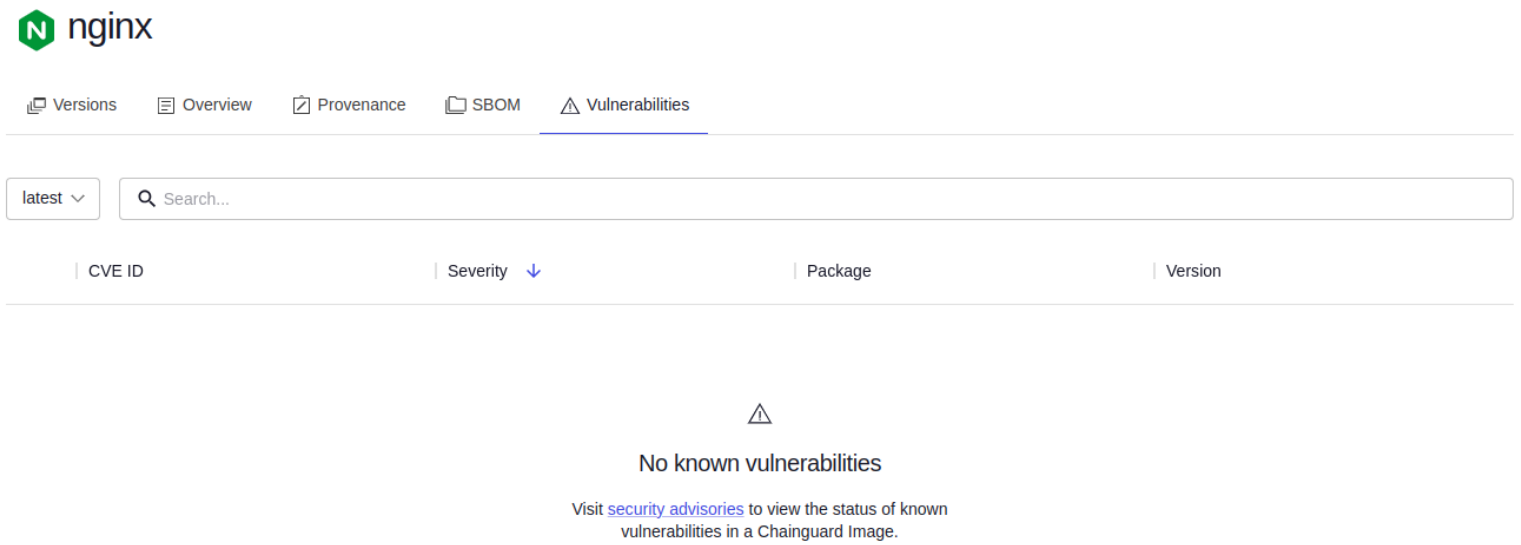


https://goharbor.io/docs/2.11.0/administration/vulnerability-scanning/

Chainguard

Цель проекта минимзация поверхности атаки.

> Minimal, hardened images with SBOMs and signatures



https://www.chainguard.dev/chainguard-images

Chainguard (Wolfi)

Собственный дистрибутив с собственной экосистемой сборки

> The key features of Wolfi are:
>
> - Provides a high-quality, build-time SBOM as standard for all packages
> - Packages are designed to be granular and independent, to support minimal images
> - Uses the proven and reliable APK package format
> - Fully declarative and reproducible build system
> - Designed to support glibc and musl

https://www.chainguard.dev/unchained/introducing-wolfi-the-first-linux-un-distro

Chainguard (Grype)

name	date	low	med	high	crit	unk	tot
cgr.dev/chainguard/nginx:latest	2024-02-17 14:34:01	0	0	0	0	0	0
nginx:latest	2024-02-17 14:34:22	4	30	12	2	2	121
cgr.dev/chainguard/nginx:latest	2024-01-21 14:33:57	0	0	0	0	0	0
nginx:latest	2024-01-21 14:34:18	4	25	11	2	9	123

https://github.com/chainguard-dev/image-comparison/blob/c224245f0bfebc9cafb44179122cb094b84b6771/data.csv

Chainguard (Example)

```
learning-labs-java on  main [!] via  desktop-linux is 📦 v1.0.0 via ☕ on ☁  adrian@chainguard.dev
took 2s
❯ docker images java-maven-cg
REPOSITORY      TAG       IMAGE ID       CREATED        SIZE
java-maven-cg   latest    55e17d645b92   2 hours ago    360MB

learning-labs-java on  main [!] via  desktop-linux is 📦 v1.0.0 via ☕ on ☁  adrian@chainguard.dev
❯ docker scout quickview java-maven-cg
    i New version 1.6.4 available (installed version is 1.5.1) at https://github.com/docker/scout-cli
    ✓ SBOM of image already cached, 117 packages indexed

  Target     │  java-maven-cg:latest  │  0C  0H  0M  0L
    digest   │  55e17d645b92          │

What's Next?
  Include policy results in your quickview by supplying an organization → docker s
maven-cg --org <organization>


learning-labs-java on  main [!] via  desktop-linux is 📦 v1.0.0 via ☕ on ☁  adr
```

https://www.youtube.com/watch?v=FYOVcSv1-oY

Chainguard

Минусы

- latest
- Своя экосистема

Плюсы

- Открытая экосистема
  - Build image - apko
  - Build APK - melange
  - Package repo
- Есть описание сборки через Dockerfile
- SLSA - Supply chain Levels for Software Artifacts
- cosign

Cosign

```
- "Keyless signing" with the Sigstore public good
    Fulcio certificate authority and Rekor transparency log (default)

- Hardware and KMS signing

- Signing with a cosign generated encrypted private/public keypair

- Container Signing, Verification and Storage in an OCI registry.

- Bring-your-own PKI
```

https://github.com/sigstore/cosign

Cosign (example with keys: prepare)

```
$ cosign generate-key-pair --output-key-prefix='cosign'
```

```
$ cosign sign --key cosign.key $IMAGE
WARNING: Image reference rusdacent/hello-container:keys uses a tag,
not a digest, to identify the image to sign.
    This can lead you to sign a different image than the intended one.
    Please use a digest (example.com/ubuntu@sha256:abc123...) rather than tag
```

Cosign (example with keys: sign)

```
$ IMAGEDIGEST=$(docker manifest inspect --verbose $IMAGE | jq -r '.Descriptor.digest')

$ export IMAGE=$IMAGE@$IMAGEDIGEST

$ echo $IMAGE
rusdacent/hello-container:keys@sha256:db5. . .452
```

```
$ cosign sign --key cosign.key $IMAGE
. . .
tlog entry created with index: 136317182
Pushing signature to: index.docker.io/rusdacent/hello-container
```

https://rewanthtammana.com/sigstore-the-easy-way/cosign/sign-and-verify-with-key/

Cosign (example with keys: verify)

```
$ cosign verify --key cosign.pub $IMAGE

Verification for index.docker.io/rusdacent/hello-container@sha256:. . . --
. . .

[
  {
    "critical": {
      "identity": {
        "docker-reference": "index.docker.io/rusdacent/hello-container"
      },
      "image": {
        "docker-manifest-digest": "sha256:db5. . .452"
      },
      "type": "cosign container image signature"
    },
    "optional": {
      "Bundle": {
        "SignedEntryTimestamp": "MEYC. . .tv+4F+2GHeP7KX",
        "Payload": {
          "body": "eyJhc. . .X0=",
          "integratedTime": 1727916426,
          "logIndex": 136317182,
          "logID": "c0d2. . .801d"
        }
. . .
]
```

https://rewanthtammana.com/sigstore-the-easy-way/cosign/sign-and-verify-with-key/

Cosign (DockerHub)

TAG

sha256-db54a230a821b4be2da51acda2489e192220bd163e64ca8d47122de5d3084452.sig

Last pushed **10 minutes ago** by rusdacent

Digest	OS/ARCH
a62a665468b4	---

TAG

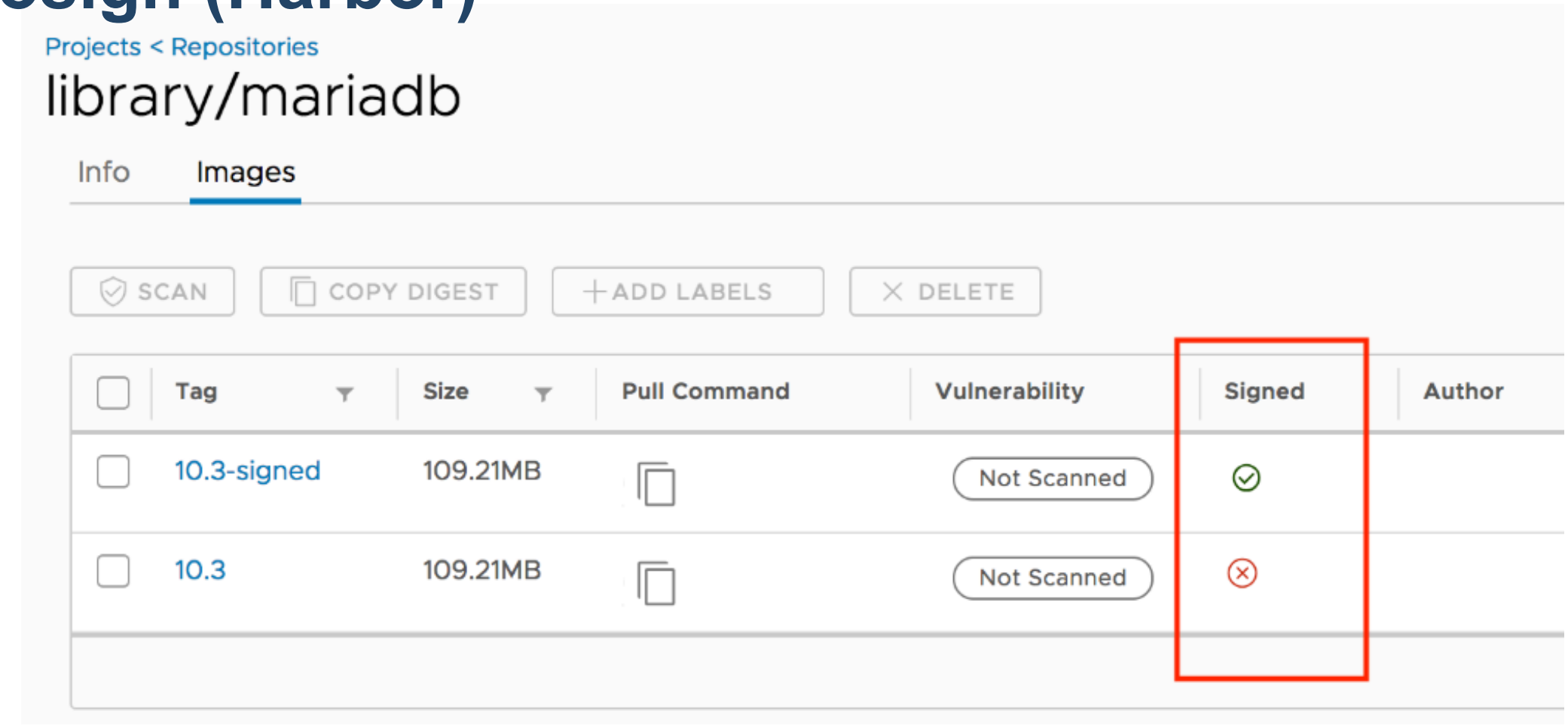
latest

Last pushed **30 minutes ago** by rusdacent

Digest	OS/ARCH
db54a230a821	linux/amd64

https://hub.docker.com/r/rusdacent/hello-container/tags

Cosign (Harbor)



https://goharbor.io/docs/2.11.0/working-with-projects/working-with-images/sign-images/

Cosign + Trivy = attestation

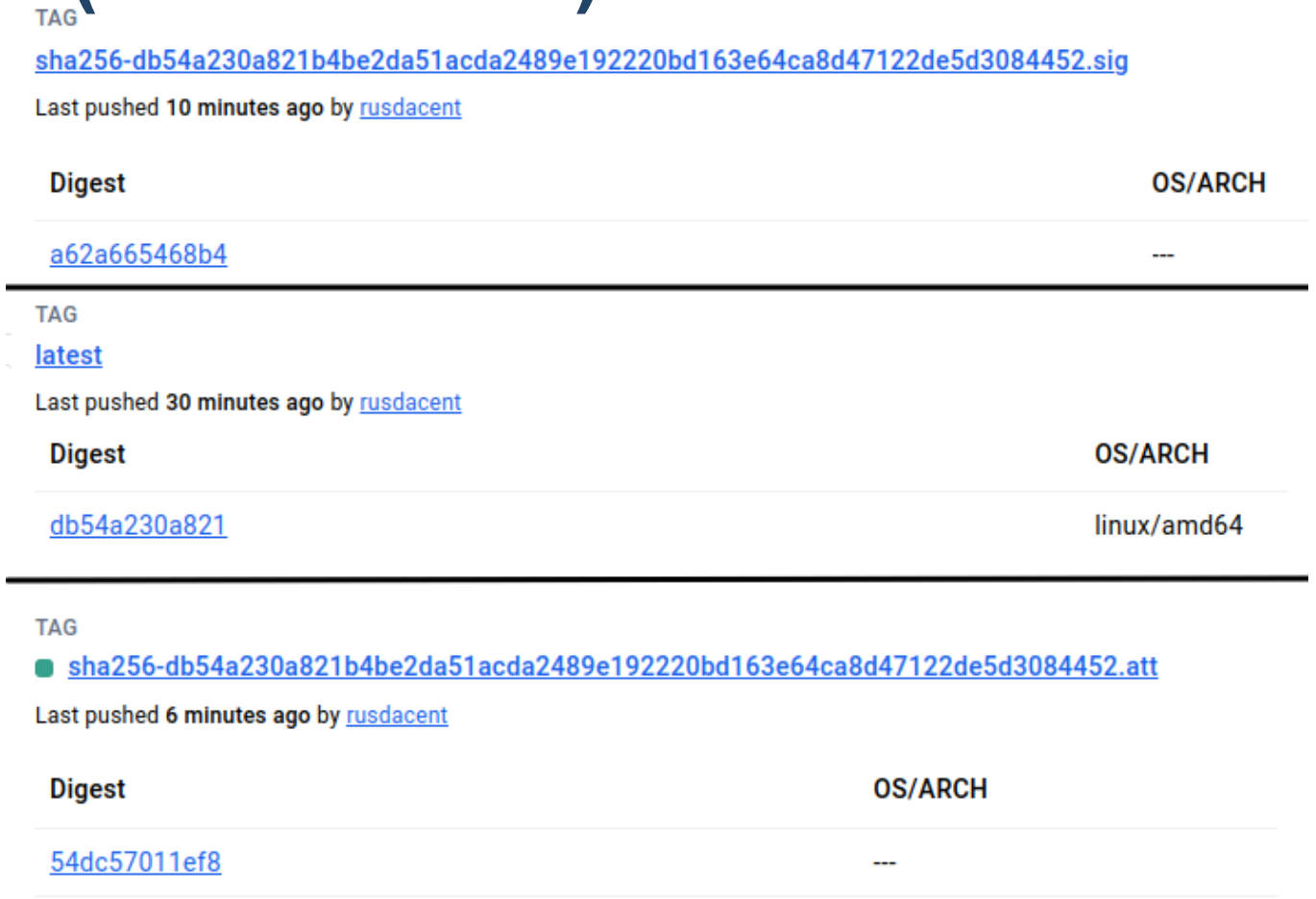
```
$ trivy image --format cosign-vuln --output vuln.json $IMG
```

```
$ cosign attest --key /path/to/cosign.key --type vuln --predicate vuln.json $IMG
```

```
$ cosign verify-attestation --key /path/to/cosign.pub --type vuln $IMG
```

https://aquasecurity.github.io/trivy/v0.56/docs/supply-chain/attestation/vuln/

Attestation (DockerHub)



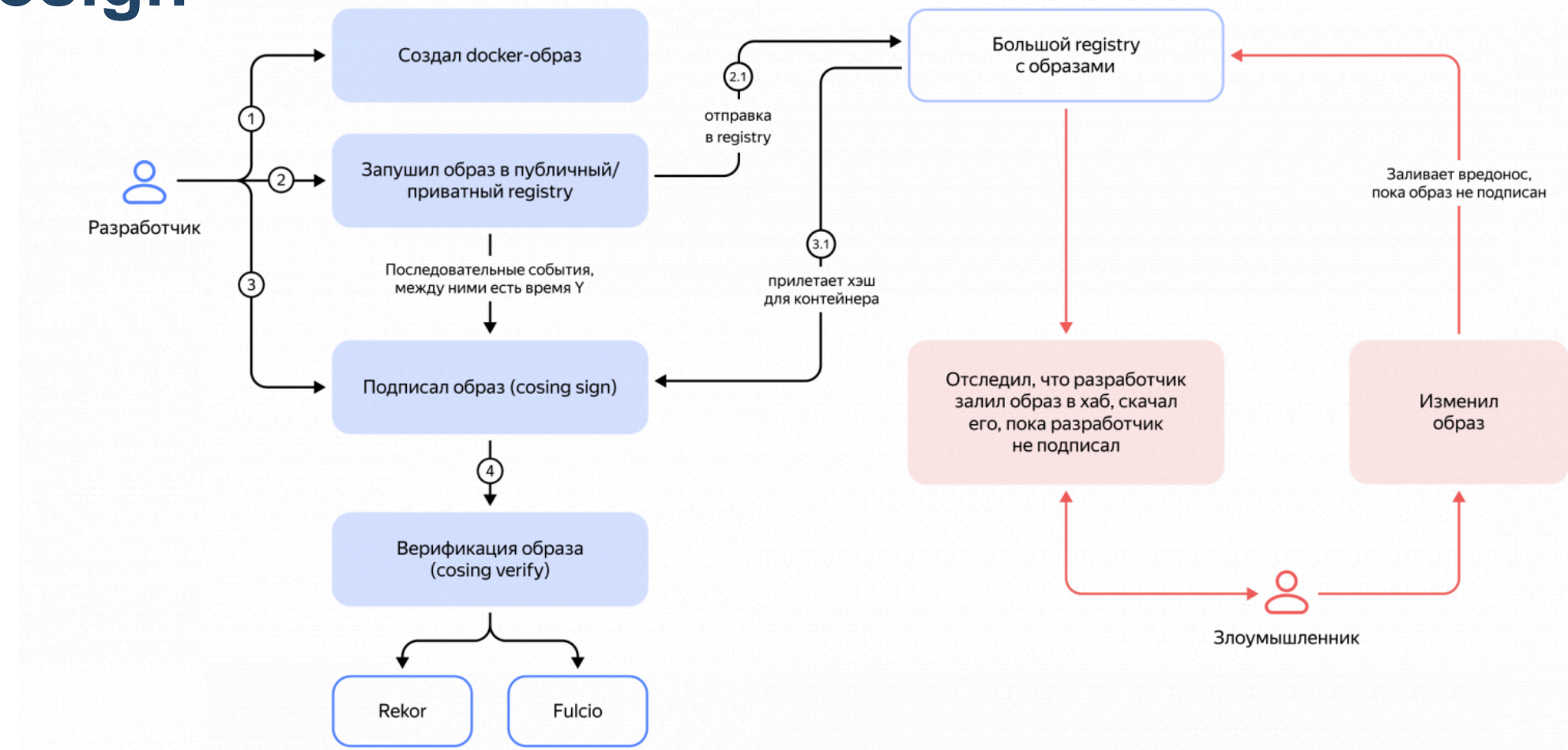
https://hub.docker.com/r/rusdacent/hello-container/tags

DockerHub + cosign

Demo

https://hub.docker.com/r/rusdacent/hello-container/tags

Cosign



https://habr.com/ru/companies/yandex_cloud_and_infra/articles/752924/

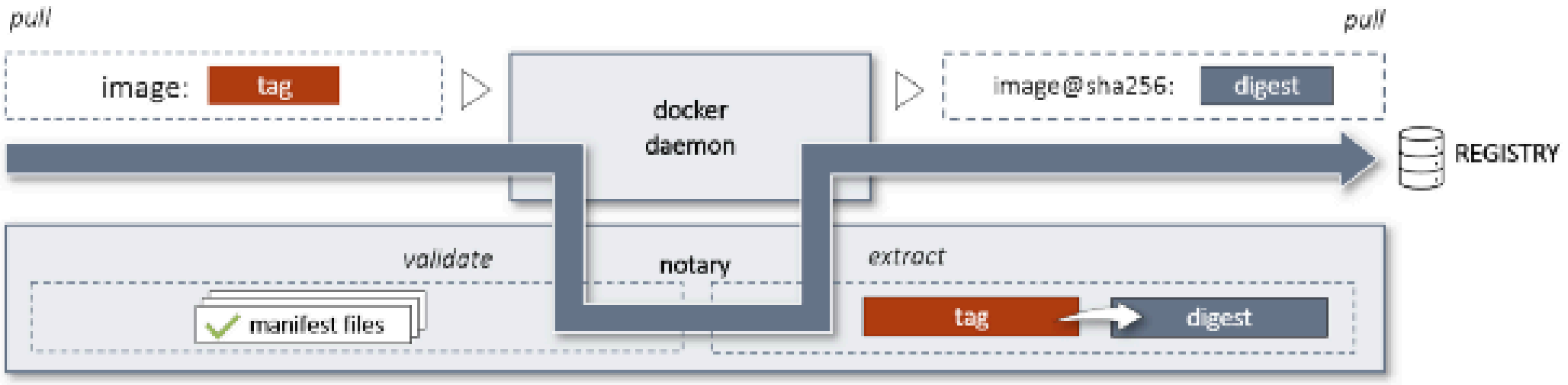
А ещё?

DCT + Notary

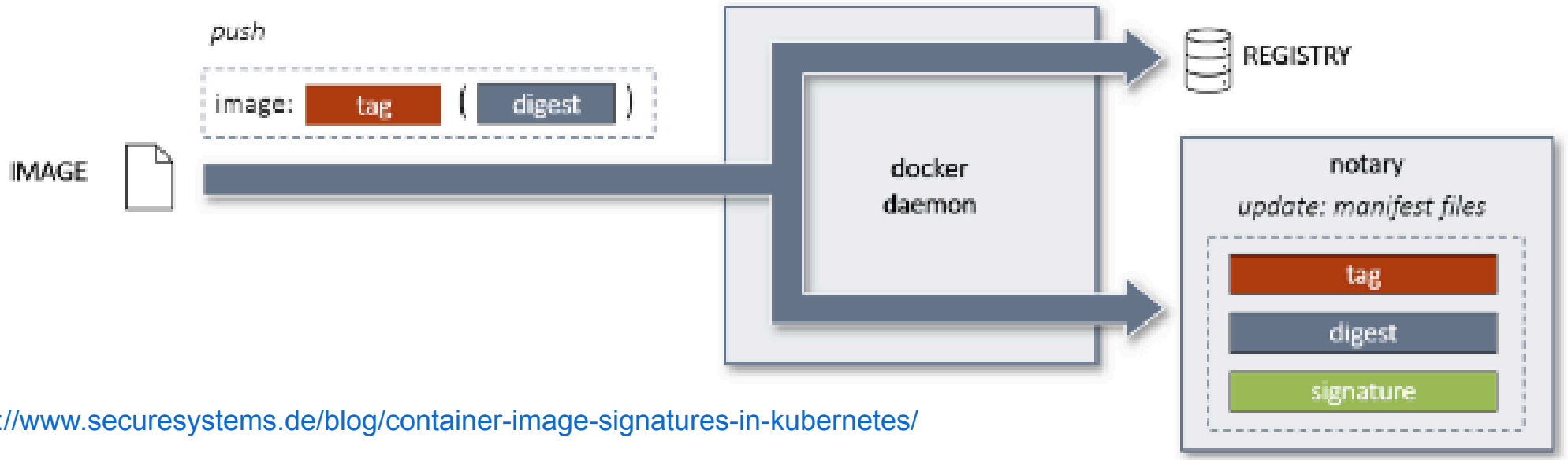
Docker Content Trust (DCT) - инструментарий, позволяющий подписывать части собираемых образов и публиковать в Notary

Notary - сервис, который хранит подписи для файлов

https://docs.docker.com/engine/security/trust/



DCT + Notary (scheme)



https://www.securesystems.de/blog/container-image-signatures-in-kubernetes/

DCT + Notary (example fail)

```
$ export DOCKER_CONTENT_TRUST=1
```

```
$ docker image pull nigelpoulton/tu-demo
Using default tag: latest

Error: remote trust data does not exist for docker.io/nigelpoulton/tu-demo:
- notary.docker.io does not have trust data for docker.io/nigelpoulton/tu-demo
```

```
$ docker image pull alpine:latest
Pull (1 of 1): alpine:latest@sha256:c5b. . .d4f27761f8e1ad6b
docker.io/library/alpine@sha256:c5b. . .d6b: Pulling from library/alpine
4abcf2066143: Pull complete
Digest: sha256:c5b. . .d6b
Status: Downloaded newer image for alpine@sha256:c5b. . .7d6b
Tagging alpine@sha256:c5b. . .d6b as alpine:latest
docker.io/library/alpine:latest
```

DCT + Notary (example success)

```
$ docker trust key generate dctkey

Successfully generated and loaded private key.
Corresponding public key available: /dirname/dctkey.pub

$ ls
dctkey.pub

$ docker push rusdacent/hello-container:dct

$ docker trust sign rusdacent/hello-container:dct
```

https://discuss.circleci.com/t/sign-docker-images-with-docker-content-trust/40319

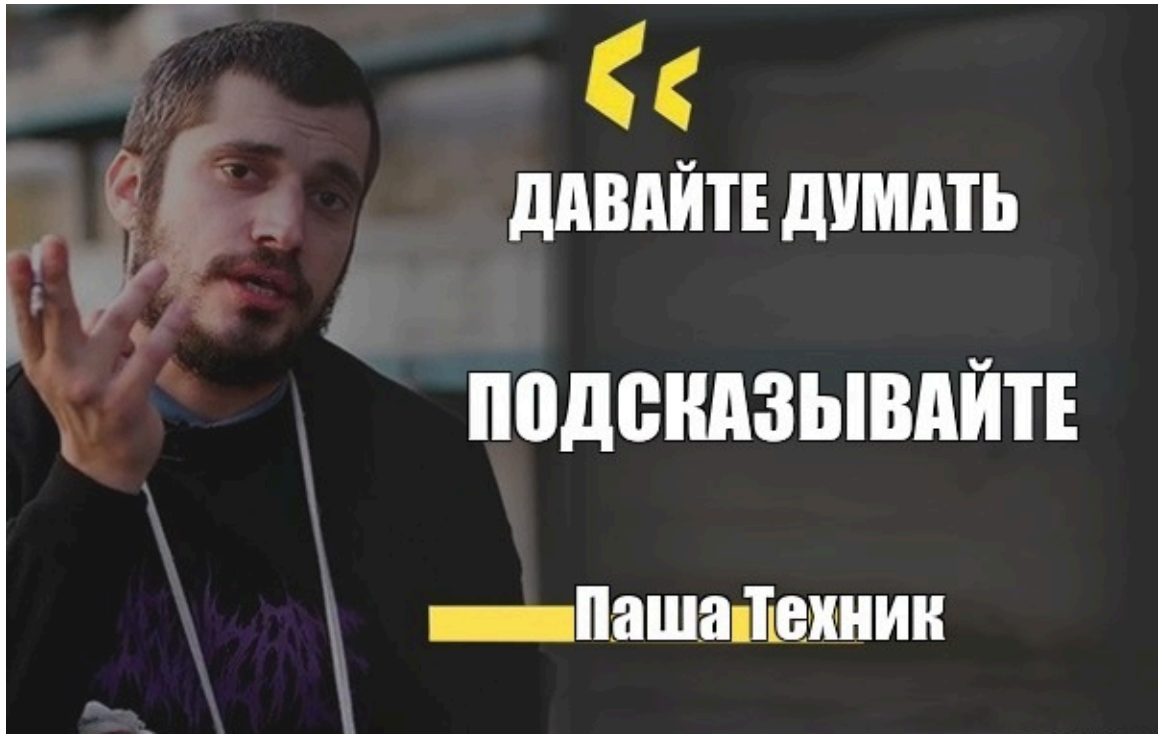
DCT + Notary (Demo)

```
$ export DOCKER_CONTENT_TRUST=1

$ docker pull rusdacent/hello-container:latest
No valid trust data for latest

$ docker pull rusdacent/hello-container:dct
Pull (1 of 1): rusdacent/hello-container:dct@sha256:db5...452
docker.io/rusdacent/hello-container@sha256:db5...452: Pulling from rusdacent/hello-container
43c4264eed91: Already exists
Digest: sha256:db5...452
Status: Downloaded newer image for rusdacent/hello-container@sha256:db5...452
Tagging rusdacent/hello-container@sha256:db5...452 as rusdacent/hello-container:dct
docker.io/rusdacent/hello-container:dct
```

Practice (Public -> Private)

Какие правила можно придумать для переноса образов во внутренний периметр?

Давайте подумаем

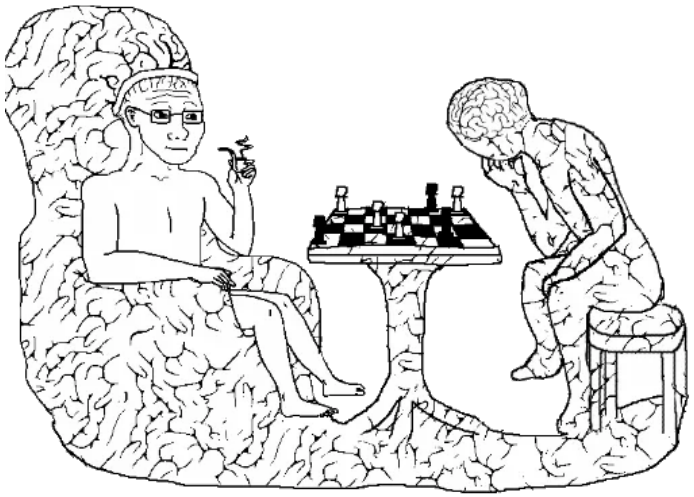


Practice (Public -> Private)

https://www.youtube.com/watch?v=biUno4LBJ3M

Какой концептуальный вывод?

Делая "SAST" не забывайте о "DAST"!



Демо пайплайн со всеми инструментами

https://gitlab.com/safecode1/safecode

Демо примеры

https://github.com/rusdacent/cooking-docker-containers/

Всем спасибо!

Вопросы?