Previously set goals

- Find the way to access the data in image and container if possible
- Try to get the running version of software
- Try to check opened ports
- Try to make your own layered image
- Logging what happens
- Check what user is running container block root
- Check for privileged mode
- Scan for Secrets
- Restricting what container can do in core system
- Intercontainer communication

To many goals we should be more focused

What we actually managed

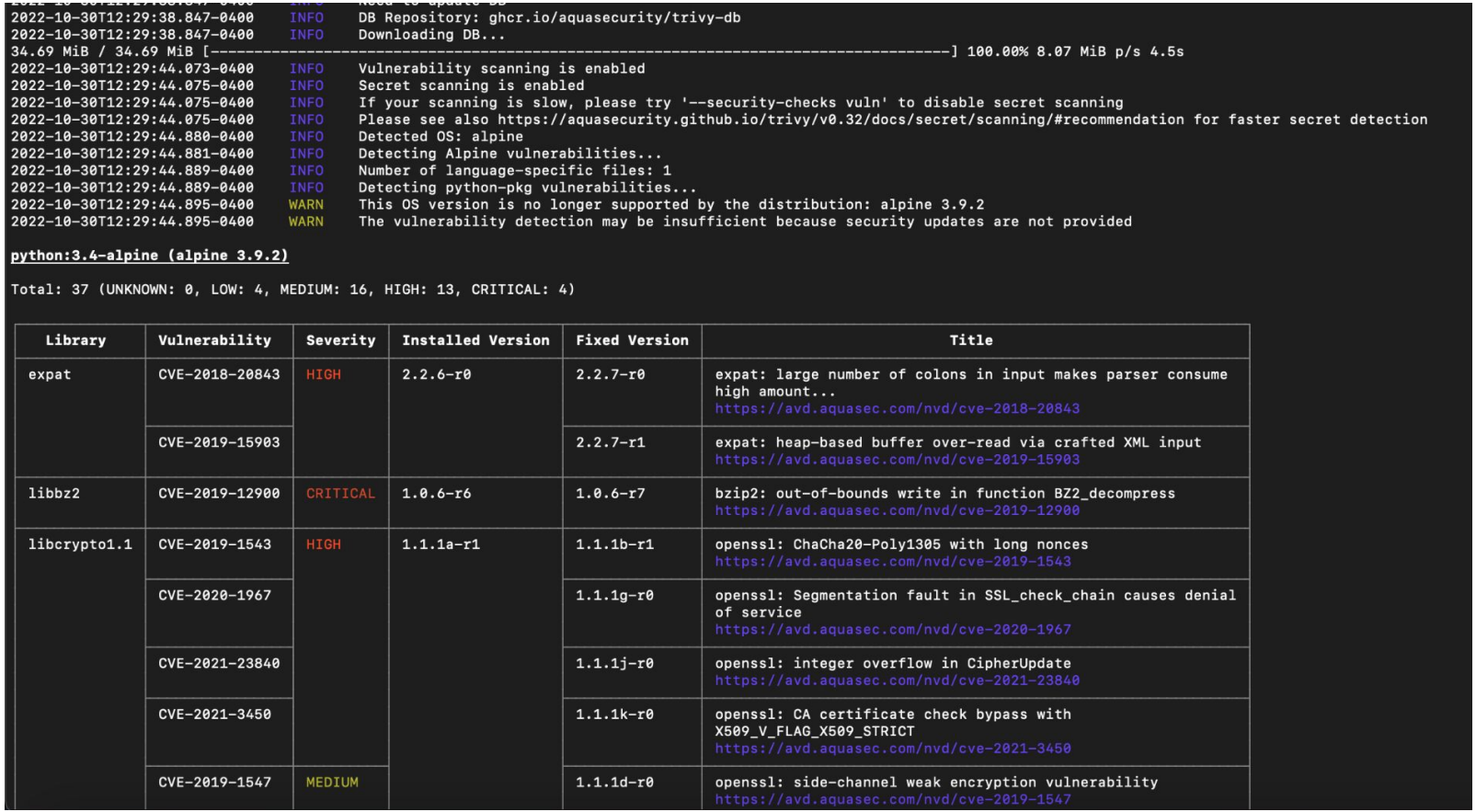
Analyzed what others do by checking cybersecurity tools (Snyke, DockerScan, Thrivy, GitGuardian) and found that:

- All of them use **CVE** database
- Software is overcomplicated

Started writing script to check containers on **local** computer (**DockerFile**, **Docker-compose**)

Found CVE **API**

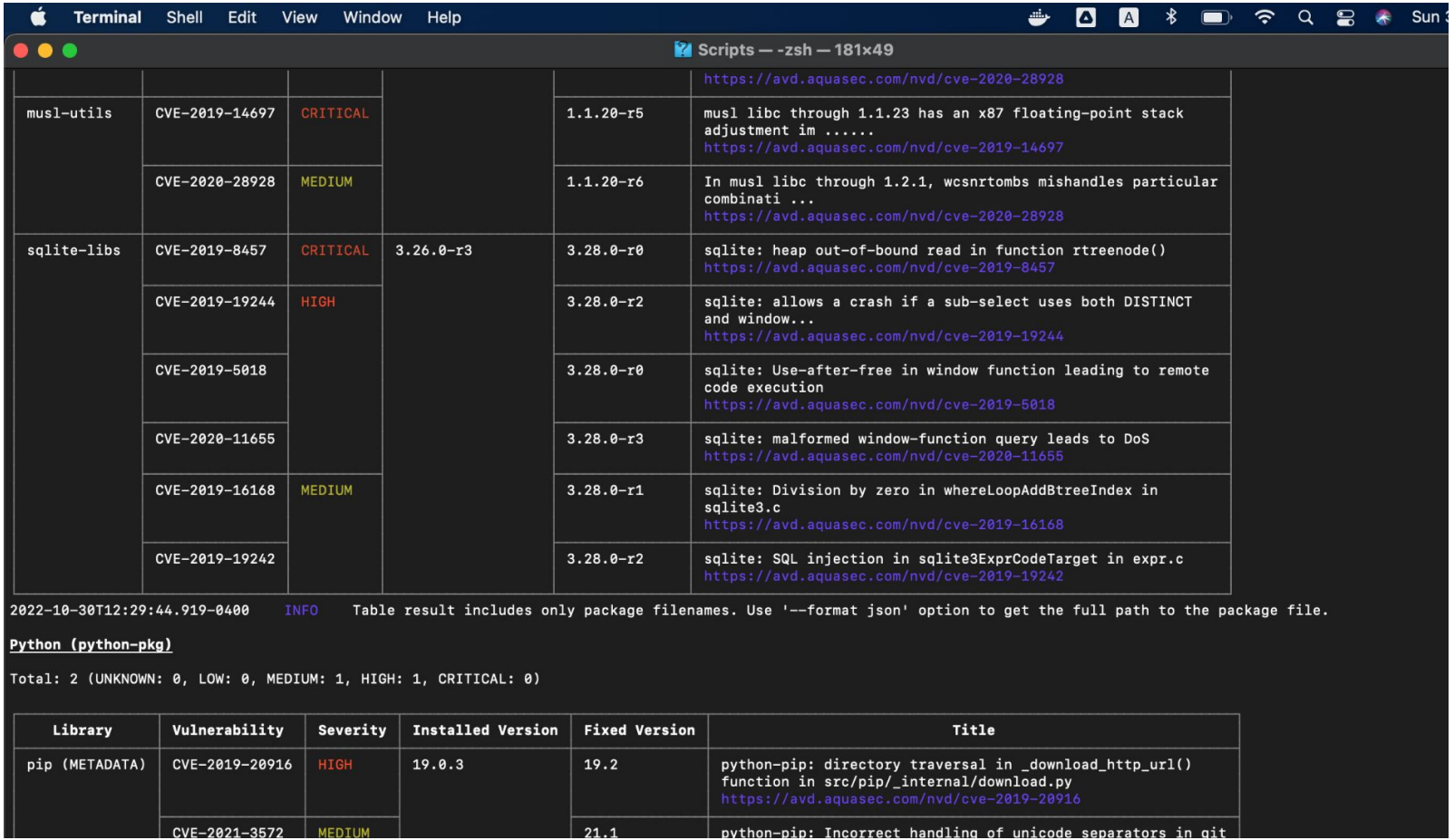
Example of scanning results open-source tool **Trivy**



```
2022-10-30T12:29:38.847-0400    INFO    DB Repository: ghcr.io/aquasecurity/trivy-db
2022-10-30T12:29:38.847-0400    INFO    Downloading DB...
34.69 MiB / 34.69 MiB [-------------------------------------------------------------------------------] 100.00% 8.07 MiB p/s 4.5s
2022-10-30T12:29:44.073-0400    INFO    Vulnerability scanning is enabled
2022-10-30T12:29:44.075-0400    INFO    Secret scanning is enabled
2022-10-30T12:29:44.075-0400    INFO    If your scanning is slow, please try '--security-checks vuln' to disable secret scanning
2022-10-30T12:29:44.075-0400    INFO    Please see also https://aquasecurity.github.io/trivy/v0.32/docs/secret/scanning/#recommendation for faster secret detection
2022-10-30T12:29:44.880-0400    INFO    Detected OS: alpine
2022-10-30T12:29:44.881-0400    INFO    Detecting Alpine vulnerabilities...
2022-10-30T12:29:44.889-0400    INFO    Number of language-specific files: 1
2022-10-30T12:29:44.889-0400    INFO    Detecting python-pkg vulnerabilities...
2022-10-30T12:29:44.895-0400    WARN    This OS version is no longer supported by the distribution: alpine 3.9.2
2022-10-30T12:29:44.895-0400    WARN    The vulnerability detection may be insufficient because security updates are not provided

python:3.4-alpine (alpine 3.9.2)

Total: 37 (UNKNOWN: 0, LOW: 4, MEDIUM: 16, HIGH: 13, CRITICAL: 4)
```

Library	Vulnerability	Severity	Installed Version	Fixed Version	Title
expat	CVE-2018-20843	HIGH	2.2.6-r0	2.2.7-r0	expat: large number of colons in input makes parser consume high amount... https://avd.aquasec.com/nvd/cve-2018-20843
	CVE-2019-15903			2.2.7-r1	expat: heap-based buffer over-read via crafted XML input https://avd.aquasec.com/nvd/cve-2019-15903
libbz2	CVE-2019-12900	CRITICAL	1.0.6-r6	1.0.6-r7	bzip2: out-of-bounds write in function BZ2_decompress https://avd.aquasec.com/nvd/cve-2019-12900
libcrypto1.1	CVE-2019-1543	HIGH	1.1.1a-r1	1.1.1b-r1	openssl: ChaCha20-Poly1305 with long nonces https://avd.aquasec.com/nvd/cve-2019-1543
	CVE-2020-1967			1.1.1g-r0	openssl: Segmentation fault in SSL_check_chain causes denial of service https://avd.aquasec.com/nvd/cve-2020-1967
	CVE-2021-23840			1.1.1j-r0	openssl: integer overflow in CipherUpdate https://avd.aquasec.com/nvd/cve-2021-23840
	CVE-2021-3450			1.1.1k-r0	openssl: CA certificate check bypass with X509_V_FLAG_X509_STRICT https://avd.aquasec.com/nvd/cve-2021-3450
	CVE-2019-1547	MEDIUM		1.1.1d-r0	openssl: side-channel weak encryption vulnerability https://avd.aquasec.com/nvd/cve-2019-1547

Results of Trivy long and difficult to understand



Terminal Shell Edit View Window Help — Sun

Scripts — -zsh — 181×49

					https://avd.aquasec.com/nvd/cve-2020-28928
musl-utils	CVE-2019-14697	CRITICAL		1.1.20-r5	musl libc through 1.1.23 has an x87 floating-point stack adjustment im https://avd.aquasec.com/nvd/cve-2019-14697
	CVE-2020-28928	MEDIUM		1.1.20-r6	In musl libc through 1.2.1, wcsnrtombs mishandles particular combinati ... https://avd.aquasec.com/nvd/cve-2020-28928
sqlite-libs	CVE-2019-8457	CRITICAL	3.26.0-r3	3.28.0-r0	sqlite: heap out-of-bound read in function rtreenode() https://avd.aquasec.com/nvd/cve-2019-8457
	CVE-2019-19244	HIGH		3.28.0-r2	sqlite: allows a crash if a sub-select uses both DISTINCT and window... https://avd.aquasec.com/nvd/cve-2019-19244
	CVE-2019-5018			3.28.0-r0	sqlite: Use-after-free in window function leading to remote code execution https://avd.aquasec.com/nvd/cve-2019-5018
	CVE-2020-11655			3.28.0-r3	sqlite: malformed window-function query leads to DoS https://avd.aquasec.com/nvd/cve-2020-11655
	CVE-2019-16168	MEDIUM		3.28.0-r1	sqlite: Division by zero in whereLoopAddBtreeIndex in sqlite3.c https://avd.aquasec.com/nvd/cve-2019-16168
	CVE-2019-19242			3.28.0-r2	sqlite: SQL injection in sqlite3ExprCodeTarget in expr.c https://avd.aquasec.com/nvd/cve-2019-19242

2022-10-30T12:29:44.919-0400 INFO Table result includes only package filenames. Use '--format json' option to get the full path to the package file.

Python (python-pkg)

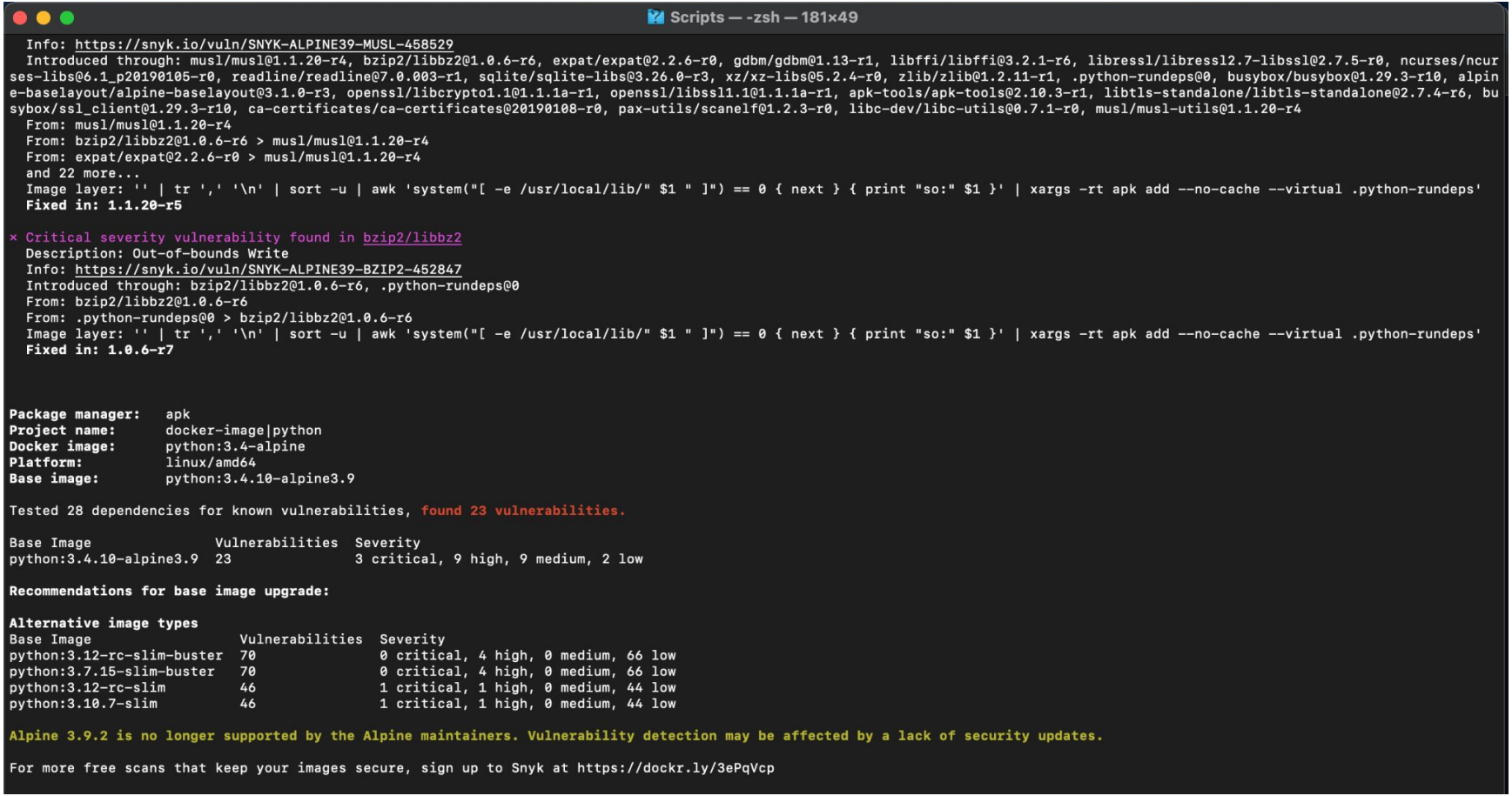
Total: 2 (UNKNOWN: 0, LOW: 0, MEDIUM: 1, HIGH: 1, CRITICAL: 0)

Library	Vulnerability	Severity	Installed Version	Fixed Version	Title
pip (METADATA)	CVE-2019-20916	HIGH	19.0.3	19.2	python-pip: directory traversal in _download_http_url() function in src/pip/_internal/download.py https://avd.aquasec.com/nvd/cve-2019-20916
	CVE-2021-3572	MEDIUM		21.1	python-pip: Incorrect handling of unicode separators in git

Example of scanning results of payd software **Snyke**

```
(base) timurzhunusov@crc-dot1x-nat-10-239-46-40 Scripts % docker scan python:3.4-alpine

Testing python:3.4-alpine...

× Low severity vulnerability found in openssl/libcrypto1.1
  Description: Inadequate Encryption Strength
  Info: https://snyk.io/vuln/SNYK-ALPINE39-OPENSSL-1089236
  Introduced through: openssl/libcrypto1.1@1.1.1a-r1, openssl/libssl1.1@1.1.1a-r1, apk-tools/apk-tools@2.10.3-r1, libtls-standalone/libtls-standalone@2.7.4-r6, ca-certificates/ca-certificates@20190108-r0
  From: openssl/libcrypto1.1@1.1.1a-r1
  From: openssl/libssl1.1@1.1.1a-r1 > openssl/libcrypto1.1@1.1.1a-r1
  From: apk-tools/apk-tools@2.10.3-r1 > openssl/libcrypto1.1@1.1.1a-r1
  and 5 more...
  Image layer: Introduced by your base image (python:3.4.10-alpine3.9)
  Fixed in: 1.1.1j-r0

× Low severity vulnerability found in openssl/libcrypto1.1
  Description: Use of a Broken or Risky Cryptographic Algorithm
  Info: https://snyk.io/vuln/SNYK-ALPINE39-OPENSSL-505098
  Introduced through: openssl/libcrypto1.1@1.1.1a-r1, openssl/libssl1.1@1.1.1a-r1, apk-tools/apk-tools@2.10.3-r1, libtls-standalone/libtls-standalone@2.7.4-r6, ca-certificates/ca-certificates@20190108-r0
  From: openssl/libcrypto1.1@1.1.1a-r1
  From: openssl/libssl1.1@1.1.1a-r1 > openssl/libcrypto1.1@1.1.1a-r1
  From: apk-tools/apk-tools@2.10.3-r1 > openssl/libcrypto1.1@1.1.1a-r1
  and 5 more...
  Image layer: Introduced by your base image (python:3.4.10-alpine3.9)
  Fixed in: 1.1.1d-r0

× Medium severity vulnerability found in sqlite/sqlite-libs
  Description: Divide By Zero
  Info: https://snyk.io/vuln/SNYK-ALPINE39-SQLITE-487067
  Introduced through: sqlite/sqlite-libs@3.26.0-r3, .python-rundeps@0
  From: sqlite/sqlite-libs@3.26.0-r3
  From: .python-rundeps@0 > sqlite/sqlite-libs@3.26.0-r3
  Image layer: '' | tr ',' '\n' | sort -u | awk 'system("[ -e /usr/local/lib/" $1 " ]") == 0 { next } { print "so:" $1 }' | xargs -rt apk add --no-cache --virtual .python-rundeps'
  Fixed in: 3.28.0-r1

× Medium severity vulnerability found in sqlite/sqlite-libs
  Description: NULL Pointer Dereference
  Info: https://snyk.io/vuln/SNYK-ALPINE39-SQLITE-587452
  Introduced through: sqlite/sqlite-libs@3.26.0-r3, .python-rundeps@0
  From: sqlite/sqlite-libs@3.26.0-r3
  From: .python-rundeps@0 > sqlite/sqlite-libs@3.26.0-r3
  Image layer: '' | tr ',' '\n' | sort -u | awk 'system("[ -e /usr/local/lib/" $1 " ]") == 0 { next } { print "so:" $1 }' | xargs -rt apk add --no-cache --virtual .python-rundeps'
  Fixed in: 3.28.0-r2
```

Results are long and difficult to read to



Comparison of results of scan

Tool	Critical	High	Medium	Low
Trivy	4	13	16	4
Snyke	3	9	9	2
Docker Skan	3	9	9	2

They are using same **CVE** (Common Vulnerabilities and Exposures) database but treat results differently choosing one over another so we are going to use it too

Our main Focus

Keep it simple and focus only on what matters for beginners following a guideline:

- Keep Docker container Up to Date
- Do Not Expose the Docker container network
- Run Docker in Rootless Mode
- Avoid Privileged Containers
- Don’t Leak Sensitive Info to Docker Images

Python Script

Right now based on **dockerfile** and **docker-compose** files we check if local container follows best practices guidelines

```
(base) timurzhunusov@crc-dot1x-nat-10-239-46-40 ~ % /Users/timurzhunusov/opt/anaconda3/bin/python /Users/timurzhunusov/Documents/GitHub/EC601/Scripts/NCIS.py
User is not defined running as root!
No exposed ports
Running software: httpd 2.4.50
Possible CVE Vulnerabilities:
[{'deprecated': False, 'cpe23Uri': 'cpe:2.3:a:apache:http_server:2.4.50:*:*:*:*:*:*:*', 'lastModifiedDate': '2021-10-13T23:19Z', 'titles': [{'title': 'Apache Software Fou
ndation Apache HTTP Server 2.4.50', 'lang': 'en_US'}], 'refs': [{'ref': 'https://httpd.apache.org/security/vulnerabilities_24.html', 'type': 'Project'}, {'ref': 'https://
us-cert.cisa.gov/ncas/current-activity/2021/10/07/apache-releases-http-server-version-2451-address-vulnerabilities', 'type': 'Advisory'}], 'deprecatedBy': [], 'vulnerabil
ities': [], 'title': 'Apache Software Foundation Apache HTTP Server 2.4.50', 'name': 'cpe:2.3:a:apache:http_server:2.4.50:*:*:*:*:*:*:*'}]
(base) timurzhunusov@crc-dot1x-nat-10-239-46-40 ~ %
```

Can check for root user, Version of software
Can connect to Free CVE to check vulnerabilities

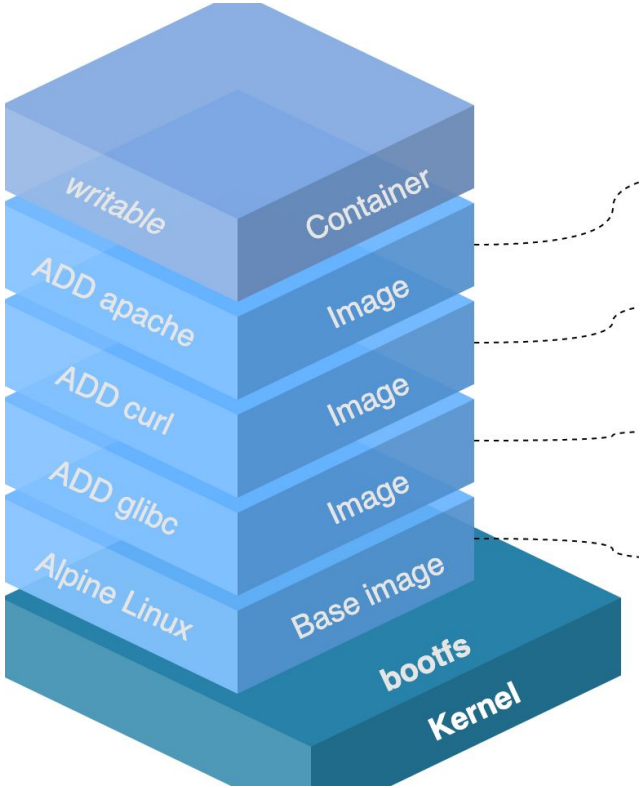
Python Script 2

For outside images like docker hub first we have to download whole image than by using **log** and **history** check what is installed
Example of **history log:**

```
(base) timurzhunusov@crc-dot1x-nat-10-239-46-40 ~ % /Users/timurzhunusov/opt/anaconda3/bin/python /Users/timurzhunusov/Documents/GitHub/EC601/Scripts/NCIS.py
IMAGE          CREATED       CREATED BY                                      SIZE      COMMENT
f05c8762fe15   2 weeks ago   /bin/sh -c #(nop)  CMD ["python3"]              0B
<missing>      2 weeks ago   /bin/sh -c set -eux;   wget -O get-pip.py "$…   10.9MB
<missing>      2 weeks ago   /bin/sh -c #(nop)  ENV PYTHON_GET_PIP_SHA256…   0B
<missing>      2 weeks ago   /bin/sh -c #(nop)  ENV PYTHON_GET_PIP_URL=ht…   0B
<missing>      2 weeks ago   /bin/sh -c #(nop)  ENV PYTHON_SETUPTOOLS_VER…   0B
<missing>      2 weeks ago   /bin/sh -c #(nop)  ENV PYTHON_PIP_VERSION=22…   0B
<missing>      2 weeks ago   /bin/sh -c set -eux;  for src in idle3 pydoc…   32B
<missing>      2 weeks ago   /bin/sh -c set -eux;   wget -O python.tar.xz…   57.1MB
<missing>      2 weeks ago   /bin/sh -c #(nop)  ENV PYTHON_VERSION=3.10.8    0B
<missing>      3 weeks ago   /bin/sh -c #(nop)  ENV GPG_KEY=A035C8C19219B…   0B
<missing>      3 weeks ago   /bin/sh -c set -eux;  apt-get update;  apt-g…   18.5MB
<missing>      3 weeks ago   /bin/sh -c #(nop)  ENV LANG=C.UTF-8             0B
<missing>      3 weeks ago   /bin/sh -c #(nop)  ENV PATH=/usr/local/bin:/…   0B
<missing>      3 weeks ago   /bin/sh -c set -ex;  apt-get update;  apt-ge…   529MB
<missing>      3 weeks ago   /bin/sh -c apt-get update && apt-get install…   152MB
<missing>      3 weeks ago   /bin/sh -c set -ex;  if ! command -v gpg > /…   19MB
<missing>      3 weeks ago   /bin/sh -c set -eux;  apt-get update;  apt-g…   10.7MB
<missing>      3 weeks ago   /bin/sh -c #(nop)  CMD ["bash"]                 0B
<missing>      3 weeks ago   /bin/sh -c #(nop) ADD file:d1268789456d2cdac…   124MB
[{'deprecated': False, 'cpe23Uri': 'cpe:2.3:a:apache:http_server:2.4.50:*:*:*:*:*:*:*', 'lastModifiedDate': '2021-10-13T23:19Z', 'titles': [{'title': 'Apache S
ndation Apache HTTP Server 2.4.50', 'lang': 'en_US'}], 'refs': [{'ref': 'https://httpd.apache.org/security/vulnerabilities_24.html', 'type': 'Project'}, {'ref'
us-cert.cisa.gov/ncas/current-activity/2021/10/07/apache-releases-http-server-version-2451-address-vulnerabilities', 'type': 'Advisory'}], 'deprecatedBy': [],
```

How to get inside the image

- Most of docker images are created on top of a base image. The base image usually from the Docker Hub.
- Some companies have their own images repositories.
- Most junior and intermediate developers doesn’t have the resources to build their own repositories. So, they typically use images downloading from public repositories such as docker hub.
- But It is possible that those images have public known vulnerabilities, attackers could abuse to gain access to their container.

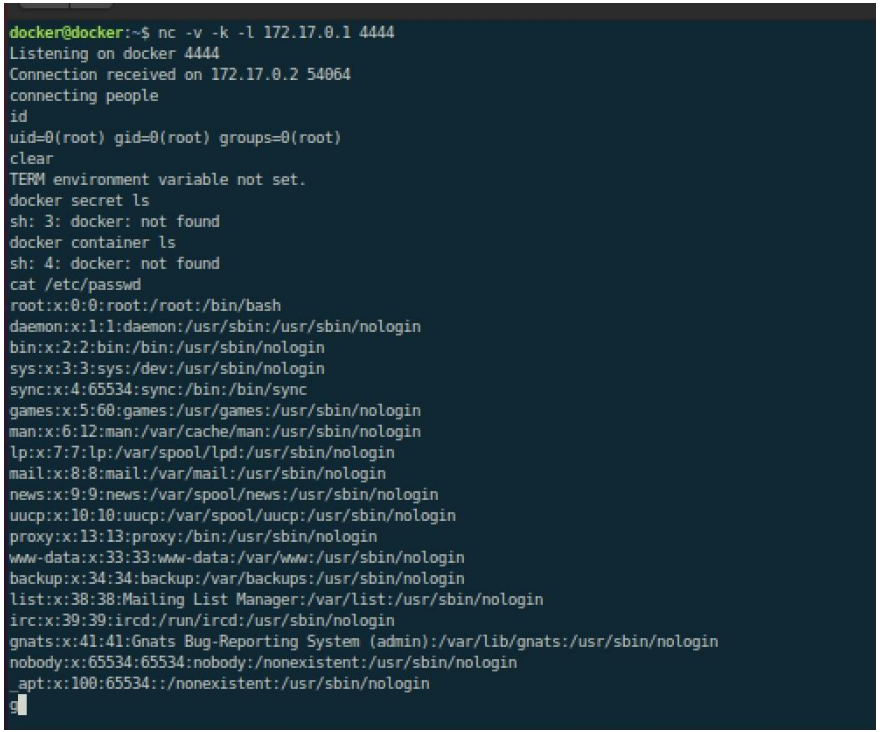


Adding a Backdoor to a docker Image

Tool: dockerscan

- Attackers download legitimate images from public repositories. They trojanizing legitimate images and re-upload them to dockhub.

- If users use those backdoored images. Attackers will gain access to their containers.



Ways to get Secrets

- Secrets usually stored in environment variables or within source code.

- If secrets are not protected properly. Anyone who can access the containers can use this property to read secrets easily.

Future goals

1. Learn to parse CVE API (might use different API because the one we using is limited)
2. Get inside the container to check for secrets
3. Check what ports are used and which ports are opened but not used