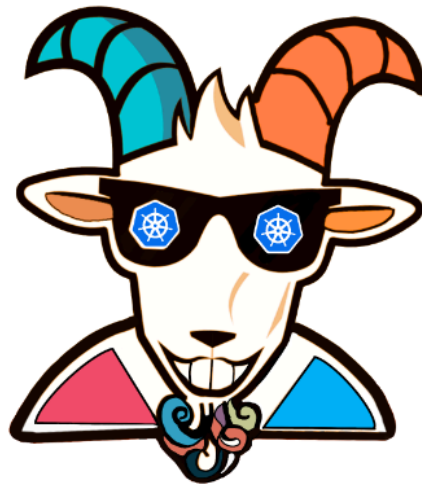


# \* Hidden in layers



## Overview

Most of the container images download and used on the internet are created by someone else. If we don't know how they get created (which means if we don't have `Dockerfile`) then I think we might be in trouble sometimes. In this scenario, we can see how to analyze the docker container image layers by using the built-in utilities and also some popular open-source utilities like `dive` to understand the container image layers.



Kubernetes Goat: Scenario diagram WIP

By the end of the scenario, we will understand and learn the following

1. How to explore, introspect and analyze docker container images
2. Using open source tools like `dive` to perform container image analysis
3. Able to work with standard command-line utilities

## ⚡ The story

Sensitive information disclosure is one of the most common vulnerabilities existing in the wild. Mishandling of passwords, private keys, tokens, etc in the containerization world is easy. Here in this scenario, we will analyze and identify one of such mishandled bad practices that leads to sensitive information disclosure.

#### ! INFO

- To get started with the scenario, run the following command and explore the `hidden-layers` job

```
kubectl get jobs
```

```
root@ip-172-31-17-86:~# kubectl get jobs
NAME                    COMPLETIONS   DURATION   AGE
hidden-in-layers       0/1            17s        17s
root@ip-172-31-17-86:~#
```

## Goal

#### 💡 TIP

Find the `k8s_goat_flag` flag value in one of the hidden layers of the container, then you have completed this scenario.

## ☐ Hints & Spoilers

▶  Still trying to understand layers?

▶  Want to get the flag?

## Solution & Walkthrough

#### ! INFO

Try exploring all files, environment variables, etc in the running container. Next, try to analyze the image used above with different tools to find exposed sensitive information.

- Docker CLI is an amazing tool with lots of features, let's start with inspecting the image

```
docker inspect madhuakula/k8s-goat-hidden-in-layers
```

```
root@ip-172-31-17-86:/tmp# docker inspect madhuakula/k8s-goat-hidden-in-layers
[
  {
    "Id": "sha256:d26427eaa7188859f2ba980fb9e746a4abd6a88c1abac99d28824f0f05156fe9",
    "RepoTags": [
      "madhuakula/k8s-goat-hidden-in-layers:latest"
    ],
    "RepoDigests": [
      "madhuakula/k8s-goat-hidden-in-layers@sha256:eb2920f014691e1b8563f09c81721f23dff7128bb70cc53583142e585feb9696"
    ],
    "Parent": "",
    "Comment": "",
    "Created": "2021-04-19T15:29:19.778391291Z",
    "Container": "e3fc82a0c3fe9d868918bb96dd699828a5867b802018b9e39058db0ded26c21b",
    "ContainerConfig": {
      "Hostname": "e3fc82a0c3fe",
      "Domainname": "",
      "User": "",
      "AttachStdin": false,
      "AttachStdout": false,
      "AttachStderr": false,
      "Tty": false,
      "OpenStdin": false,
      "StdinOnce": false,
      "Env": [
        "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
      ],
      "Cmd": [
        "/bin/sh",
        "-c",
        "#(nop) ",
        "CMD [\"sh\" \"-c\" \"tail -f /dev/null\"]"
      ],
      "Image": "sha256:8196f8da57a41c09ae47dec/cf9817d9c/3a8824dae5cb1dbfc94aaa939a7584",

```

- We can observe the `cmd` section in the above output. It shows the default command executed by this image on startup. Though this shows some interesting information, it's not good enough for us
- Maybe it would be more helpful for us if we get to know how this image is built from scratch. For that, we need to analyze the `Dockerfile` of the image. If you have `Dockerfile`, it's good. If not, there are a few ways to analyze it

## Method 1

- We can explore each layer by using the default `docker history` command

```
docker history --no-trunc madhuakula/k8s-goat-hidden-in-layers
```

```
root@ip-172-31-17-86:/tmp# docker history --no-trunc madhuakula/k8s-goat-hidden-in-layers
IMAGE                                SIZE      COMMENT      CREATED      CREATED BY
sha256:d26427eaa7188859f2ba980fb9e746a4abd6a88c1abac99d28824f0f05156fe9  24 hours ago  /bin/sh -c #(nop) CMD [\"sh\" \"-c\" \"tail -f /dev/null\"]
<missing>                             24 hours ago  /bin/sh -c echo \"Contributed by Rewanth Cool\" >> /root/contributio
n.txt && rm -rf /root/secret.txt      28B
<missing>                             24 hours ago  /bin/sh -c #(nop) ADD file:828b1f78e37f3be0c1aadd7b6e84634ff7f3049
1b11f2461a4d4f8bfbf5ad64c in /root/secret.txt  41B
<missing>                             24 hours ago  /bin/sh -c #(nop) LABEL MAINTAINER=Madhu Akula INFO=Kubernetes Go
at
<missing>                             6 months ago  /bin/sh -c #(nop) CMD [\"/bin/sh\"]
<missing>                             6 months ago  /bin/sh -c #(nop) ADD file:f17f65714f703db9012f00e5ec98d0b2541ff61
47c2633f7ab9ba659d0c507f4 in /
root@ip-172-31-17-86:/tmp#
```

## Method 2

- We can use the `dfimage` to generate a Dockerfile of any given image. First, we can set up that and perform it by running the following commands

```
alias dfimage="docker run -v /var/run/docker.sock:/var/run/docker.sock --rm alpine/dfimage"
```

```
dfimage -sV=1.36 madhuakula/k8s-goat-hidden-in-layers
```

```
root@ip-172-31-17-86:/tmp# alias dfimage="docker run -v /var/run/docker.sock:/var/run/docker.sock --rm alpine/dfimage"
root@ip-172-31-17-86:/tmp# dfimage -sV=1.36 madhuakula/k8s-goat-hidden-in-layers
Analyzing madhuakula/k8s-goat-hidden-in-layers
Docker Version: 20.10.6
GraphDriver: overlay2
Environment Variables
|PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin

Image user
|User is root

Potential secrets:
|Found match etc/apk/keys/alpine-devel@lists.alpinelinux.org-4a6a0840.rsa.pub Possible public key \.pub$ ff72598b05f57e6f83d56b858ba9783796e99aea2dcef391b3fa688c1e077ae5/layer.tar
|Found match etc/apk/keys/alpine-devel@lists.alpinelinux.org-5243ef4b.rsa.pub Possible public key \.pub$ ff72598b05f57e6f83d56b858ba9783796e99aea2dcef391b3fa688c1e077ae5/layer.tar
|Found match etc/apk/keys/alpine-devel@lists.alpinelinux.org-5261cecb.rsa.pub Possible public key \.pub$ ff72598b05f57e6f83d56b858ba9783796e99aea2dcef391b3fa688c1e077ae5/layer.tar
|Found match etc/udhcpd.conf DHCP server configs dhcpd[^]*.conf ff72598b05f57e6f83d56b858ba9783796e99aea2dcef391b3fa688c1e077ae5/layer.tar
Dockerfile:
CMD ["/bin/sh"]
LABEL MAINTAINER=Madhu_Akula INFO=Kubernetes_Goat
ADD file:828b1f78e37f3be0c1aadd7b6e84634ff7f30491b11f2461a4d4f8bfbf5ad64c in /root/secret.txt
    root/
    root/secret.txt

RUN echo "Contributed by Rewanath Cool" >> /root/contribution.txt \
    && rm -rf /root/secret.txt
CMD ["sh" "-c" "tail -f /dev/null"]

root@ip-172-31-17-86:/tmp#
```

## Method 3

- `dive` is an amazing tool that helps with analyzing each layer of an image

Layers			Current Layer Contents			Filetree
Cmp	Size	Command	Permission	UID:GID	Size	
5.6 MB	FROM	ff72598b05f57e6	-rw-----	0:0	0 B	lock
41 B	#(nop)	ADD file:828b1f78e37f3be0c1aadd7b6e84634ff7f30491b11f2461a4d	-rw-r--r--	0:0	11 kB	scripts.tar
28 B	echo	"Contributed by Rewanath Cool" >> /root/contribution.txt &&	-rw-r--r--	0:0	76 B	triggers
Layer Details			drwxr-xr-x	0:0	0 B	firmware
Tags: (unavailable)			-rwxr-xr-x	0:0	596 kB	ld-musl-x86_64.so.1
Id: da73da4359e9edb793ee5472ae3538be8aec57c27efff7dae8873566c865533f			-rwxrwxrwx	0:0	0 B	libc.musl-x86_64.so.1 → ld-musl-x86_64.so.1
Digest: sha256:602cf959640172877386041de1a9c7a4baaa11093b566b03663537d7bab0643e			-rwxr-xr-x	0:0	2.6 MB	libcrypto.so.1.1
Command: #(nop) ADD file:828b1f78e37f3be0c1aadd7b6e84634ff7f30491b11f2461a4d4f8bfbf5ad64			-rwxr-xr-x	0:0	524 kB	libssl.so.1.1
			-rwxrwxrwx	0:0	0 B	libz.so.1 → libz.so.1.2.11
			-rwxr-xr-x	0:0	100 kB	libz.so.1.2.11
			drwxr-xr-x	0:0	0 B	mdev
			drwxr-xr-x	0:0	0 B	modules-load.d
			drwxr-xr-x	0:0	1.3 kB	sysctl.d
			-rw-r--r--	0:0	1.3 kB	00-alpine.conf
			drwxr-xr-x	0:0	0 B	media
			drwxr-xr-x	0:0	0 B	cdrom
			drwxr-xr-x	0:0	0 B	floppy
			drwxr-xr-x	0:0	0 B	usb
			drwxr-xr-x	0:0	0 B	mnt
			drwxr-xr-x	0:0	0 B	opt
			dr-xr-xr-x	0:0	0 B	proc
			drwx-----	0:0	41 B	root
			-rw-rw-r--	0:0	41 B	secret.txt
			drwxr-xr-x	0:0	0 B	run
			drwxr-xr-x	0:0	226 kB	sbin
			-rwxrwxrwx	0:0	0 B	acpid → /bin/busybox
			-rwxrwxrwx	0:0	0 B	adjtimex → /bin/busybox
			-rwxr-xr-x	0:0	211 kB	apk
			-rwxrwxrwx	0:0	0 B	arp → /bin/busybox
			-rwxrwxrwx	0:0	0 B	blkid → /bin/busybox
			-rwxrwxrwx	0:0	0 B	blockdev → /bin/busybox
			-rwxrwxrwx	0:0	0 B	depmod → /bin/busybox
			-rwxrwxrwx	0:0	0 B	fb splash → /bin/busybox
			-rwxrwxrwx	0:0	0 B	fdisk → /bin/busybox
			-rwxrwxrwx	0:0	0 B	findfs → /bin/busybox
			-rwxrwxrwx	0:0	0 B	fscck → /bin/busybox
			-rwxrwxrwx	0:0	0 B	fstrim → /bin/busybox
			-rwxrwxrwx	0:0	0 B	getty → /bin/busybox

## NOTE

From all the above analyses, we can see some significant changes in these two files, `/root/contributions.txt`, `/root/secret.txt`. The above methods cannot help us to read the contents of these files. Let's see if we can find these files in the running container.

```
root@ip-172-31-17-86:~# kubectl run hello --rm --restart=Never -it --image=madhuakula/k8s-goat-hidden-in-layers -- sh
If you don't see a command prompt, try pressing enter.
/ # ls /root -la
total 16
drwx----- 1 root    root          4096 Apr 20 16:06 .
drwxr-xr-x  1 root    root          4096 Apr 20 16:06 ..
-rw-----  1 root    root           13 Apr 20 16:06 .ash_history
-rw-r--r--  1 root    root          28 Apr 19 15:29 contribution.txt
/ # cat /root/contribution.txt
Contributed by Rewanath Cool
/ # cat /root/secret.txt
cat: can't open '/root/secret.txt': No such file or directory
/ #
```

- We can't see `/root/secret.txt` as it is deleted from the next layers. We can recover the `/root/secret.txt` by leveraging the docker built-in command to export the docker image as a tar file

```
docker save madhuakula/k8s-goat-hidden-in-layers -o hidden-in-layers.tar
```

- Now we have the artifact and we can extract the tar file to explore the layers

```
tar -xvf hidden-in-layers.tar
```

```
root@ip-172-31-17-86:/tmp/hidden-in-layers# ls
hidden-in-layers.tar
root@ip-172-31-17-86:/tmp/hidden-in-layers# tar xvf hidden-in-layers.tar
d26427eaa7188859f2ba980fb9e746a4abd6a88c1abac99d28824f0f05156fe9.json
d81b961e28e91cdb9ea742b87bdc69685a6c4c76368a0a3de0934584a992e051/
d81b961e28e91cdb9ea742b87bdc69685a6c4c76368a0a3de0934584a992e051/VERSION
d81b961e28e91cdb9ea742b87bdc69685a6c4c76368a0a3de0934584a992e051/json
d81b961e28e91cdb9ea742b87bdc69685a6c4c76368a0a3de0934584a992e051/layer.tar
da73da4359e9edb793ee5472ae3538be8aec57c27efff7dae8873566c865533f/
da73da4359e9edb793ee5472ae3538be8aec57c27efff7dae8873566c865533f/VERSION
da73da4359e9edb793ee5472ae3538be8aec57c27efff7dae8873566c865533f/json
da73da4359e9edb793ee5472ae3538be8aec57c27efff7dae8873566c865533f/layer.tar
ff72598b05f57e6f83d56b858ba9783796e99aea2dcef391b3fa688c1e077ae5/
ff72598b05f57e6f83d56b858ba9783796e99aea2dcef391b3fa688c1e077ae5/VERSION
ff72598b05f57e6f83d56b858ba9783796e99aea2dcef391b3fa688c1e077ae5/json
ff72598b05f57e6f83d56b858ba9783796e99aea2dcef391b3fa688c1e077ae5/layer.tar
manifest.json
repositories
root@ip-172-31-17-86:/tmp/hidden-in-layers#
```

#### TIP

We can see each layer getting exported as a single tar file. We have 3 layers in this image, so we have 3 tar files. Since we have only 3 layers, it's easy to extract all of them and check the contents but that's not the conventional approach. What if we have hundreds of layers?

- Let's review the dive output again. In the below image, we saw a new file, `/root/secret.txt` is being created

Layers	Current Layer Contents	Size	Filetree
Cmp	Permission	UID:GID	
5.6 MB	FROM ff72598b05f57e6	0 B	lock
41 B	#(nop) ADD file:828b1f78e37f3be0c1aadd7b6e84634ff7f30491b11f2461a4d	11 kB	scripts.tar
28 B	echo "Contributed by Rewanthee Cool" >> /root/contribution.txt &&	76 B	triggers
Layer Details			firmware
Tags:			ld-musl-x86_64.so.1
Id:	da73da4359e9edb793ee5472ae3538be8aec57c27efff7dae8873566c865533f	596 kB	libc.musl-x86_64.so.1 → ld-musl-x86_64.so.1
Digest:	sha256:602cf959640172877386041de1a9c7a4baaa11093b566b03663537d7bab0643e	0 B	libcrypto.so.1.1
Command:	#(nop) ADD file:828b1f78e37f3be0c1aadd7b6e84634ff7f30491b11f2461a4d4f8bfb5ad64	524 kB	libssl.so.1.1
Image Details			libz.so.1 → libz.so.1.2.11
Image name: madhuakula/k8s-goat-hidden-in-layers			libz.so.1.2.11
Total Image size: 5.6 MB			mdev
Potential wasted space: 41 B			modules-load.d
Image efficiency score: 99 %			sysctl.d
Count	Total Space	Path	00-alpine.conf
2	41 B	/root/secret.txt	media
			cdrom
			floppy
			usb
			mnt
			opt
			proc
			root
			secret.txt
			run
			sbin
			acpid → /bin/busybox
			adjtimex → /bin/busybox
			apk
			arp → /bin/busybox
			blkid → /bin/busybox
			blockdev → /bin/busybox
			depmod → /bin/busybox
			fb splash → /bin/busybox
			fdisk → /bin/busybox
			findfs → /bin/busybox
			fscck → /bin/busybox
			fstrim → /bin/busybox
			getty → /bin/busybox

- Observe the **Id** of that layer, `da73da4359e9edb793ee5472ae3538be8aec57c27efff7dae8873566c865533f`. Since, we have `/root/secret.txt` created in this layer, let's extract the tar file of this layer first

```
cd da73da4359e9edb793ee5472ae3538be8aec57c27efff7dae8873566c865533f
```

```
tar -xvf layer.tar
```

```
cat root/secret.txt
```

```
root@ip-172-31-17-86:/tmp/hidden-in-layers# ls
d26427eaa7188859f2ba980fb9e746a4abd6a88c1abac99d28824f0f05156fe9.json ff72598b05f57e6f83d56b858ba9783796e99aea2dcef391b3fa688c1e077ae5 repositories
d81b961e28e91c9b9ea742b7bdc69685a6c4c76368a0a3de0934584a992e051 hidden-in-layers.tar
da73da4359e9edb793ee5472ae3538be8aec57c27efff7dae8873566c865533f manifest.json
root@ip-172-31-17-86:/tmp/hidden-in-layers# cd da73da4359e9edb793ee5472ae3538be8aec57c27efff7dae8873566c865533f
root@ip-172-31-17-86:/tmp/hidden-in-layers/da73da4359e9edb793ee5472ae3538be8aec57c27efff7dae8873566c865533f# ls
VERSION json layer.tar
root@ip-172-31-17-86:/tmp/hidden-in-layers/da73da4359e9edb793ee5472ae3538be8aec57c27efff7dae8873566c865533f# tar xvf layer.tar
root/
root/secret.txt
root@ip-172-31-17-86:/tmp/hidden-in-layers/da73da4359e9edb793ee5472ae3538be8aec57c27efff7dae8873566c865533f# cat root/secret.txt
k8s-goat-3b7a7dc7f51f4014ddf3446c25f8b772root@ip-172-31-17-86:/tmp/hidden-in-layers/da73da4359e9edb793ee5472ae3538be8aec57c27efff7dae8873566c865533f#
```

- Hooray 🥳, now we found Kubernetes Goat flag



## References

- Digging into docker layers
- dive

