Docker image journey

How to shrink a docker image

DevOps Pro Europe 2019
2019-03-20

MOOVEL BECOMES

REACHNOW

pic: © Moovel

pic: © Moovel

Me

Engineering Manager

github.com/lotharschulz

@lothar_schulz

lotharschulz.info

speakerdeck.com/lothar

pic: https://pixabay.com/en/turtle-tortoise-swim-sea-turtle-863336

pic: https://www.flickr.com/photos/eyecatcherfotosde/41584801594

Why reduce docker image size ?

- Pulling and pushing docker images from and to remote docker registries is faster

- Security attack surface is often smaller

Proof ?

Well ...

Team Journey part 1

```
FROM golang:1.12

RUN mkdir /app
WORKDIR /app
COPY hello.go .

RUN go build -o hellogo .

RUN groupadd -g 99 appuser && useradd -r -u 99 -g appuser appuser
USER appuser

CMD ["./hellogo"]
```

```
$ docker build --rm -t ...
```

https://github.com/lotharschulz/hellogodocker/blob/master/Dockerfile

Team Journey part 1

Image size: 780 MB (compressed 299 MB)

CircleCI build time: ~ 0.5 sec

Dockerhub push time: ~ 4 sec

Dockerhub pull time: ~ 21 sec

https://circleci.com/gh/lotharschulz/hellogodocker/105

pull to empty nodes, no layers cached

Team Journey part 1.1

```
FROM golang:1.12

RUN mkdir /app
WORKDIR /app
COPY hello.go .

RUN go build -o hellogo .

RUN groupadd -g 99 appuser && useradd -r -u 99 -g appuser appuser
USER appuser

CMD ["./hellogo"]
```

```
$ docker build --cache-from golang:1.12 -t ...
```

https://github.com/lotharschulz/hellogodocker/blob/master/Dockerfile

Team Journey part 1.1

previous step

Image size compressed: 780 MB (c: 299)

CircleCI build time: ~ 0.5 sec

Dockerhub push time: ~ 4 sec

Dockerhub pull time: ~ 21 sec

https://circleci.com/gh/lotharschulz/hellogodocker/105

current step

Image size compressed: 780 MB (c: 299)

CircleCI build time: ~ 0,6 sec

Dockerhub push time: ~ 4 sec

Dockerhub pull time: ~ 18 sec

cache warm up in preparation step

Team Journey part 2 - builder pattern

```
#part one
FROM golang:1.12 as builder
# workdir setup
COPY hello.go    .

RUN CGO_ENABLED=0 GOOS=linux go build -a -installsuffix cgo -ldflags '-w -extldflags
"-static"' -o hellogodocker .

# part two
FROM alpine:latest
# user & workdir setup ….
COPY --from=builder /go/src/github.com/lotharschulz/hellogodocker/hellogodocker .

CMD ["./hellogodocker"]
```

```
$ docker build --rm -t ...
```

https://github.com/lotharschulz/hellogodocker/blob/master/DockerfileBuilder

please note the static linked binary

Team Journey part 2 - builder pattern

previous step		current step
Image size compressed: **780 MB**	→	Image size compressed: **11.9 MB** (c: 5)
CircleCI build time: ~ 0.6 sec		CircleCI build time: ~ 0.6 sec
Dockerhub push time: ~ 4 sec		Dockerhub push time: ~ 3.4 sec
Dockerhub pull time: **~ 18 sec**	→	Dockerhub pull time: **~ 1.2 sec**

https://circleci.com/gh/lotharschulz/hellogodocker/105

Team Journey part 2 - builder pattern - squash & compress

```
#part one
FROM golang:1.12 as builder
# workdir setup
COPY hello.go    .

RUN CGO_ENABLED=0 GOOS=linux go build -a -installsuffix cgo -ldflags '-w -extldflags
"-static"' -o hellogodocker .

# part two
FROM alpine:latest
# user & workdir setup ….
COPY --from=builder /go/src/github.com/lotharschulz/hellogodocker/hellogodocker .

CMD ["./hellogodocker"]
```

```
$ docker build --squash/--compress --rm -t ...
```

https://github.com/lotharschulz/hellogodocker/blob/master/DockerfileBuilder

squash is an experimental feature

Team Journey part 2 - builder pattern - squash & compress

previous step	→	current step
Image size compressed: 11.9 MB (c: 5)		Image size compressed: 11.9 MB (c: 5)
CircleCI build time: ~ 0.6 sec		CircleCI build time: ~ 0.6/1.1 sec
Dockerhub push time: ~ 3.4 sec		Dockerhub push time: ~ 3.5/4.1 sec
Dockerhub pull time: ~ 1.2 sec		Dockerhub pull time: ~ 1.1/1.3 sec

https://circleci.com/gh/lotharschulz/hellogodocker/98

Team Journey part 3 - alpine base

```
FROM alpine:latest
# ... set up user

ADD hellogo /
CMD ["/hellogo"]
```

```
$ docker build [--squash/--compress] --rm -t ...
```

static linked binary created within ci worker node outside of Dockerfile scope

https://circleci.com/gh/lotharschulz/hellogodocker/105

Team Journey part 3 - alpine base

previous step	→	current step
Image size compressed: 11.9 MB (c: 5)		Image size compressed: 12.2 MB (c: 5)
CircleCI build time: ~ 0.6/1.1 sec		CircleCI build time: ~ 0.6 sec
Dockerhub push time: ~ 3.5/4.1 sec		Dockerhub push time: ~ 3.5 sec
Dockerhub pull time: ~ 1.1/1.3 sec		Dockerhub pull time: ~ 1.2 sec

https://circleci.com/gh/lotharschulz/hellogodocker/105

Team Journey part 4 - FROM scratch

```
FROM scratch
ADD ca-certificates.crt /etc/ssl/certs/
ADD hellogo /
CMD ["/hellogo"]
```

```
$ docker build [--squash/--compress] --rm -t ...
```

static linked binary created within ci worker node outside of Dockerfile scope

https://github.com/lotharschulz/hellogodocker/blob/master/DockerfileAlpine

Team Journey part 4 - FROM scratch

previous step		current step
Image size compressed: 12.2 MB	→	Image size compressed: **4.8 MB** (c: 2)
CircleCI build time: ~ 0.6 sec	→	CircleCI build time: **~ 15 sec**
Dockerhub push time: ~ 3.5 sec		Dockerhub push time: ~ 3.5 sec
Dockerhub pull time: ~ 1.2 sec		Dockerhub pull time: ~ 0.7 sec

https://circleci.com/gh/lotharschulz/hellogodocker/105

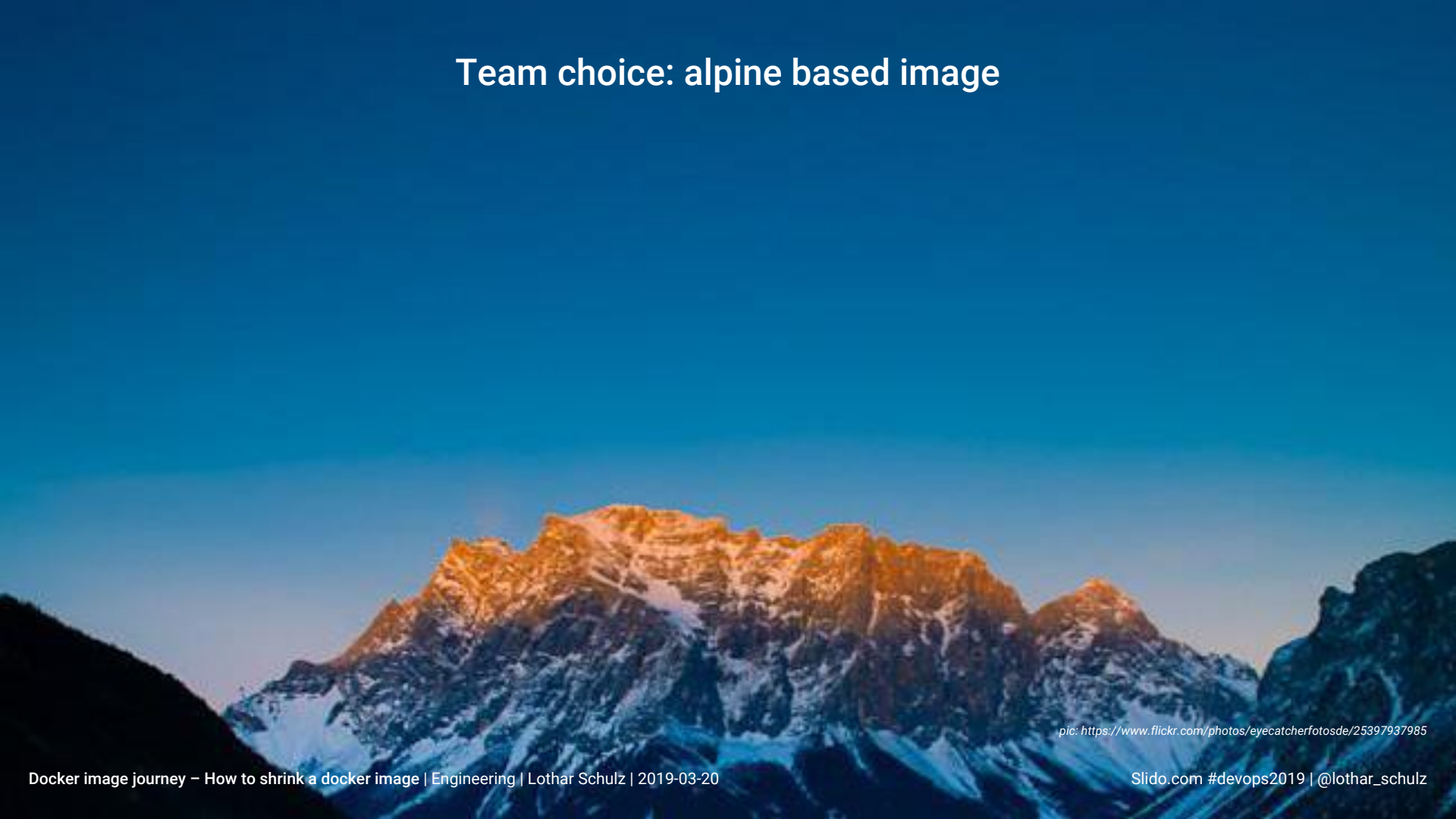
What about Ops?


```
$ docker run -p 1234:1234 --name hellogodocker lotharschulz/hellogo:build.docker-min--0.2.102
```

```
$ docker exec -it hellogodocker sh

  OCI runtime exec failed: exec failed: container_linux.go:344:

  starting container process caused "exec: \"sh\":

  executable file not found in $PATH": unknown
```

pic: https://pixabay.com/en/caution-label-warning-red-mark-943376/

Team choice: alpine based image



pic: https://www.flickr.com/photos/eyecatcherfotosde/25397937985

Some numbers

image size compressed: **780 MB**

Dockerhub pull time: **~ 21 sec**

0.6%

5%

image size compressed: **4,8 MB**

Dockerhub pull time: **~ 0.7 sec**

attack surface - FROM golang:1.12

```
CLAIR_OUTPUT=High CLAIR_ADDR=http://192.168.99.100:30060 klar
lotharschulz/hellogo:build.docker-cache--0.2.101
clair timeout 1m0s
docker timeout: 1m0s
no whitelist file
Analysing 11 layers
Got results from Clair API v1
Found 300 vulnerabilities
Unknown: 8
Negligible: 71
Low: 95
Medium: 107
High: 19
```

attack surface - FROM scratch

```
CLAIR_OUTPUT=High CLAIR_ADDR=http://192.168.99.100:30060 klar
lotharschulz/hellogo:build.docker-min--0.2.101
clair timeout 1m0s
docker timeout: 1m0s
no whitelist file
Analysing 2 layers
Got results from Clair API v1
Found 0 vulnerabilities
```

Conclusions

Balance

- docker image size
- security considerations
- infrastructural effort

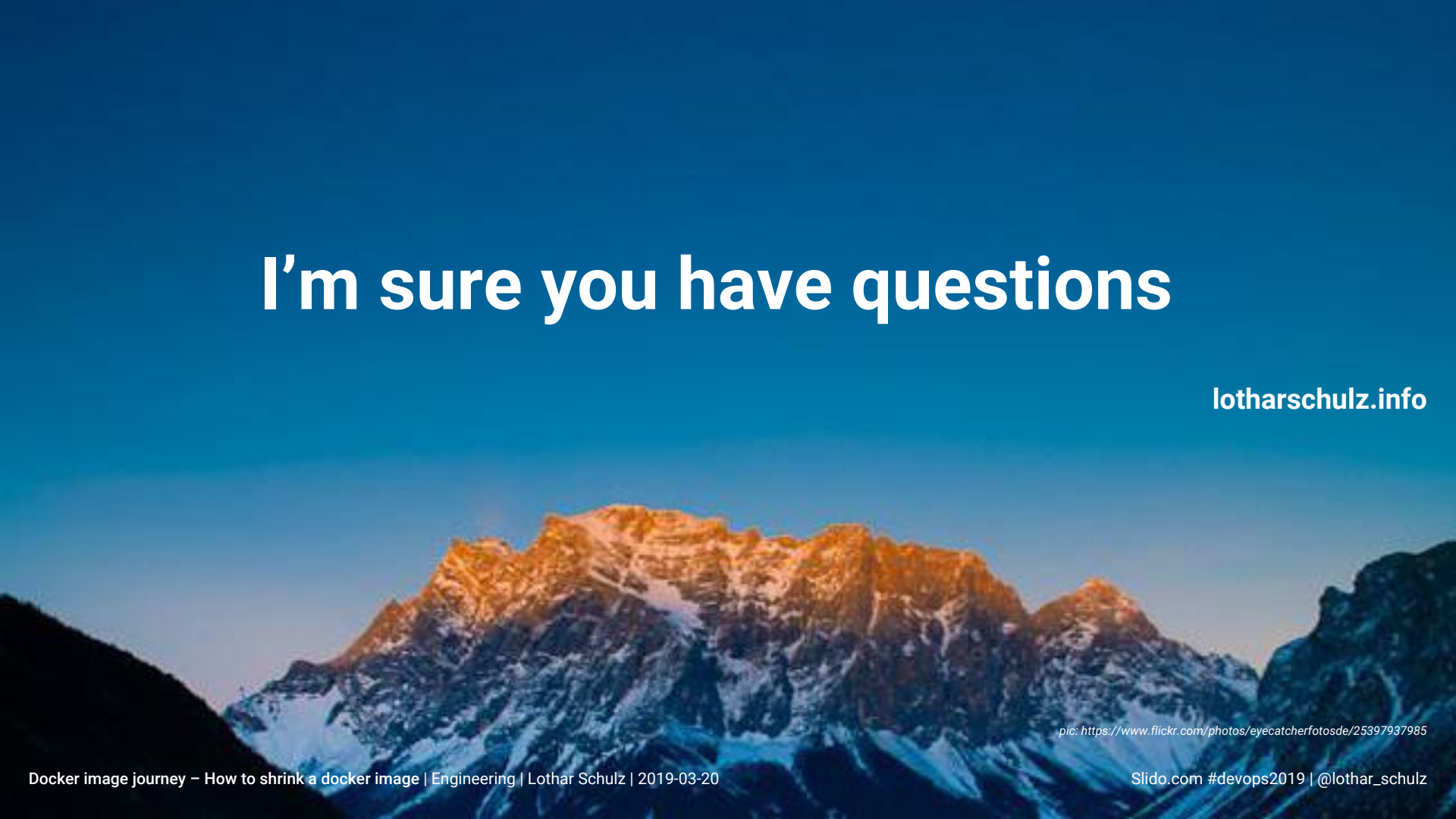
Conclusions

- potentially reduced container startup time (pull)
- potentially reduced attack surface

”Small Images are always better in terms of absolute performance”

I’m sure you have questions

lotharschulz.info



pic: https://www.flickr.com/photos/eyecatcherfotosde/25397937985

Backup

```
CLAIR_OUTPUT=High CLAIR_ADDR=http://192.168.99.100:30060 klar lotharschulz/hellogo:build.docker-cache--0.2.101    2019 03 10
clair timeout 1m0s
docker timeout: 1m0s
Docker config: 
Analyzing 11 layers
Got results from Clair API v1
Found 368 vulnerabilities
Unknown: 8
Negligible: 71
Low: 180
Medium: 107
High: 19

CVE-2018-15686: [High]
Found in: systemd [232-25+deb9u8]
Fixed By: 
A vulnerability in unit_deserialize of systemd allows an attacker to supply arbitrary state across systemd re-execution via NotifyAccess. This can be used to improperly influence systemd execution and possibly lead to root privilege escalation. Affected releases are systemd versions up to and including 239.
https://security-tracker.debian.org/tracker/CVE-2018-15686
---

CVE-2018-6954: [High]
Found in: systemd [232-25+deb9u8]
Fixed By: 
systemd-tmpfiles in systemd through 237 mishandles symlinks present in non-terminal path components, which allows local users to obtain ownership of arbitrary files via vectors involving creation of a directory and a file under that directory, and later replacing that directory with a symlink. This occurs even if the fs.protected_symlinks sysctl is turned on.
https://security-tracker.debian.org/tracker/CVE-2018-6954
---

CVE-2019-1000001: [High]
Found in: glibc [2.24-11+deb9u4]
Fixed By: 
In glibc 2.26 and earlier there is confusion in the usage of getcwd() by realpath() which can be used to write before the destination buffer leading to a buffer underflow and potential code execution.
https://security-tracker.debian.org/tracker/CVE-2019-1000001
---

CVE-2018-6485: [High]
Found in: glibc [2.24-11+deb9u4]
Fixed By: 
An integer overflow in the implementation of the posix_memalign in memalign functions in the GNU C Library (aka glibc or libc6) 2.26 and earlier could cause these functions to return a pointer to a heap area that is too small, potentially leading to heap corruption.
https://security-tracker.debian.org/tracker/CVE-2018-6485
---

CVE-2018-6551: [High]
Found in: glibc [2.24-11+deb9u4]
Fixed By: 
The malloc implementation in the GNU C Library (aka glibc or libc6), from version 2.24 to 2.26 on powerpc, and only in version 2.26 on i386, did not properly handle malloc calls with arguments close to SIZE_MAX and could return a pointer to a heap region that is smaller than requested, eventually leading to heap corruption.
https://security-tracker.debian.org/tracker/CVE-2018-6551
---

CVE-2019-9169: [High]
Found in: glibc [2.24-11+deb9u4]
Fixed By: 
In the GNU C Library (aka glibc or libc6) through 2.29, proceed_next_node in posix/regexec.c has a heap-based buffer over-read via an attempted case-insensitive regular-expression match.
https://security-tracker.debian.org/tracker/CVE-2019-9169
---

CVE-2016-2779: [High]
Found in: util-linux [2.29.2-1+deb9u1]
Fixed By: 
runuser in util-linux allows local users to escape to the parent session via a crafted TIOCSTI ioctl call, which pushes characters to the terminal's input buffer.
https://security-tracker.debian.org/tracker/CVE-2016-2779
---

CVE-2017-17458: [High]
Found in: mercurial [4.0-1+deb9u1]
Fixed By: 
In Mercurial before 4.4.1, it is possible that a specially malformed repository can cause Git subrepositories to run arbitrary code in the form of a .git/hooks/post-update script checked into the repository. Typical use of Mercurial prevents construction of such repositories, but they can be created programmatically.
https://security-tracker.debian.org/tracker/CVE-2017-17458
---

CVE-2018-13347: [High]
Found in: mercurial [4.0-1+deb9u1]
Fixed By: 
mpatch.c in Mercurial before 4.6.1 mishandles integer addition and subtraction, aka OVE-20180430-0002.
https://security-tracker.debian.org/tracker/CVE-2018-13347
---

CVE-2013-7445: [High]
Found in: linux [4.9.144-3.1]
Fixed By: 
The Direct Rendering Manager (DRM) subsystem in the Linux kernel through 4.x mishandles requests for Graphics Execution Manager (GEM) objects, which allows context-dependent attackers to cause a denial of service (memory consumption) via an application that processes graphics data, as demonstrated by JavaScript code that creates many CANVAS elements for rendering by Chrome or Firefox.
https://security-tracker.debian.org/tracker/CVE-2013-7445
---

CVE-2018-12931: [High]
Found in: linux [4.9.144-3.1]
Fixed By: 
ntfs_attr_find in the ntfs.ko filesystem driver in the Linux kernel 4.15.0 allows attackers to trigger a stack-based out-of-bounds write and cause a denial of service (kernel oops or panic) or possibly have unspecified other impact via a crafted ntfs filesystem.
https://security-tracker.debian.org/tracker/CVE-2018-12931
---

CVE-2019-7308: [High]
Found in: linux [4.9.144-3.1]
Fixed By: 
kernel/bpf/verifier.c in the Linux kernel before 4.20.6 performs undesirable out-of-bounds speculation on pointer arithmetic in various cases, including cases of different branches with different state or limits to sanitize, leading to side-channel attacks.
https://security-tracker.debian.org/tracker/CVE-2019-7308
---

CVE-2018-9517: [High]
Found in: linux [4.9.144-3.1]
Fixed By: 
In pppol2tp_connect, there is possible memory corruption due to a use after free. This could lead to local escalation of privilege with System execution privileges needed. User interaction is not needed for exploitation. Product: Android. Versions: Android kernel. Android ID: A-38159931.
https://security-tracker.debian.org/tracker/CVE-2018-9517
---

CVE-2018-12930: [High]
Found in: linux [4.9.144-3.1]
Fixed By: 
fsck_ntfs_read_buffer_async_read in the ntfs.ko filesystem driver in the Linux kernel 4.15.0 allows attackers to trigger a stack-based out-of-bounds write and cause a denial of service (kernel oops or panic) or possibly have unspecified other impact via a crafted ntfs filesystem.
https://security-tracker.debian.org/tracker/CVE-2018-12930
---

CVE-2018-20169: [High]
Found in: linux [4.9.144-3.1]
Fixed By: 
An issue was discovered in the Linux kernel before 4.19.0. The USB subsystem mishandles size checks during the reading of an extra descriptor, related to __usb_get_extra_descriptor in drivers/usb/core/usb.c.
https://security-tracker.debian.org/tracker/CVE-2018-20169
---

CVE-2017-1000379: [High]
Found in: linux [4.9.144-3.1]
Fixed By: 
The Linux Kernel running on AMD64 systems will sometimes map the contents of PIE executable, the heap or ld.so to where the stack is mapped allowing attackers to more easily manipulate the stack. Linux Kernel version 4.11.5 is affected.
https://security-tracker.debian.org/tracker/CVE-2017-1000379
---

CVE-2019-6974: [High]
Found in: linux [4.9.144-3.1]
Fixed By: 
A memory leak in the kernel_read_file function in fs/exec.c in the Linux kernel through 4.20.11 allows attackers to cause a denial of service (memory consumption) by triggering vfs_read failures.
https://security-tracker.debian.org/tracker/CVE-2019-6974
---

CVE-2017-12424: [High]
Found in: shadow [1:4.4-4.1]
Fixed By: 
In shadow before 4.5, the newusers tool could be made to manipulate internal data structures in ways unintended by the authors. Malformed input may lead to crashes (with a buffer overflow or other memory corruption) or other unspecified behaviors. This crosses a privilege boundary in, for example, certain web-hosting environments in which a Control Panel allows an unprivileged user account to create subaccounts.
https://security-tracker.debian.org/tracker/CVE-2017-12424
---

CVE-2017-14062: [High]
Found in: libidn [1.33-1]
Fixed By: 
Integer overflow in the decode_digit function in puny_decode.c in Libidn2 before 2.0.4 allows remote attackers to cause a denial of service or possibly have unspecified other impact.
https://security-tracker.debian.org/tracker/CVE-2017-14062
---
```

Team Journey part 1

```
FROM golang:1.11

RUN mkdir /app
WORKDIR /app
COPY hello.go .

RUN go build -o hellogo .

RUN groupadd -g 99 appuser && useradd -r -u 99 -g appuser appuser
USER appuser

CMD ["./hellogo"]
```

```
$ docker build --rm -t ...
```


https://github.com/lotharschulz/hellogodocker/blob/master/Dockerfile

Team Journey part 1

```
Image size compressed: 299 MB

CircleCI build time: ~ 3 sec

Dockerhub push time: ~ 4 sec

Dockerhub pull time: ~ 21 ses
```

```
pull to empty nodes, no layers cached
```


https://circleci.com/gh/lotharschulz/hellogodocker/91

Team Journey part 1.1

```
FROM golang:1.11

RUN mkdir /app
WORKDIR /app
COPY hello.go .

RUN go build -o hellogo .

RUN groupadd -g 99 appuser && useradd -r -u 99 -g appuser appuser
USER appuser

CMD ["./hellogo"]
```

```
$ docker build --cache-from golang:1.11 -t ...
```


https://github.com/lotharschulz/hellogodocker/blob/master/Dockerfile

Team Journey part 1.1

previous step

Image size compressed: 299 MB

CircleCI build time: **~ 3 sec**

Dockerhub push time: ~ 4 sec

Dockerhub pull time: ~ 21 ses

current step

Image size compressed: 299 MB

CircleCI build time: **~ 0.5 sec**

Dockerhub push time: ~ 4 sec

Dockerhub pull time: ~ 21 ses

cache warm up in preparation step

https://circleci.com/gh/lotharschulz/hellogodocker/91

Team Journey part 2 - builder pattern

```
#part one
FROM golang:1.11 as builder
# workdir setup
COPY hello.go    .

RUN CGO_ENABLED=0 GOOS=linux go build -a -installsuffix cgo -ldflags '-w -extldflags
"-static"' -o hellogodocker .

# part two
FROM alpine:latest
# user & workdir setup ….
COPY --from=builder /go/src/github.com/lotharschulz/hellogodocker/hellogodocker .

CMD ["./hellogodocker"]
```

```
$ docker build --rm -t ...
```

use a static linked binary

https://github.com/lotharschulz/hellogodocker/blob/master/DockerfileBuilder

Team Journey part 2 - builder pattern

previous step		current step
Image size compressed: **299 MB**	→	Image size compressed: **5 MB**
CircleCI build time: ~ 0.5 sec		CircleCI build time: ~ 0.6 sec
Dockerhub push time: ~ 4 sec		Dockerhub push time: ~ 4 sec
Dockerhub pull time: **~ 21 ses**	→	Dockerhub pull time: **~ 1.2 ses**

https://circleci.com/gh/lotharschulz/hellogodocker/91

Team Journey part 2 - builder pattern - squash & compress

```
#part one
FROM golang:1.11 as builder
# workdir setup
COPY hello.go    .

RUN CGO_ENABLED=0 GOOS=linux go build -a -installsuffix cgo -ldflags '-w -extldflags
"-static"' -o hellogodocker .

# part two
FROM alpine:latest
# user & workdir setup ….
COPY --from=builder /go/src/github.com/lotharschulz/hellogodocker/hellogodocker .

CMD ["./hellogodocker"]
```

```
$ docker build --squash/--compress --rm -t ...
```

squash is an experimental Docker daemon option

https://github.com/lotharschulz/hellogodocker/blob/master/DockerfileBuilder

Team Journey part 2 - builder pattern - squash & compress

previous step	→	current step
Image size compressed: 5 MB		Image size compressed: 5 MB
CircleCI build time: ~ 0.6 sec		CircleCI build time: ~ 0.7/1.1 sec
Dockerhub push time: ~ 4 sec		Dockerhub push time: ~ 4/4.9 sec
Dockerhub pull time: ~ 1.2 ses		Dockerhub pull time: ~ 1.1/1.4 ses

https://circleci.com/gh/lotharschulz/hellogodocker/91

Team Journey part 3 - alpine base

```
FROM alpine:latest
# … set up user

ADD hellogo /
CMD ["/hellogo"]
```

```
$ docker build [--squash/--compress] --rm -t ...
```

static linked binary created within ci worker node outside of Dockerfile scope

https://github.com/lotharschulz/hellogodocker/blob/master/DockerfileAlpine

Team Journey part 3 - alpine base

previous step		current step
Image size compressed: 5 MB		Image size compressed: 5 MB
CircleCI build time: ~ 0.7/1.1 sec		CircleCI build time: ~ 0.6/1.1 sec
Dockerhub push time: ~ 4/4.9 sec		Dockerhub push time: ~ 3.8/4.6 sec
Dockerhub pull time: ~ 1.1/1.4 ses		Dockerhub pull time: ~ 1.1 ses

https://circleci.com/gh/lotharschulz/hellogodocker/91

Team Journey part 4 - FROM scratch

```
FROM scratch
ADD ca-certificates.crt /etc/ssl/certs/
ADD hellogo /
CMD ["/hellogo"]
```

```
$ docker build [--squash/--compress] --rm -t ...
```

static linked binary created within ci worker node outside of Dockerfile scope

https://github.com/lotharschulz/hellogodocker/blob/master/DockerfileAlpine

Team Journey part 4 - FROM scratch

previous step		current step
Image size compressed: **5 MB**	→	Image size compressed: **2 MB**
CircleCI build time: **~ 0.6/1.1 sec**	→	CircleCI build time: **~ 16 sec**
Dockerhub push time: ~ 3.8/4.6 sec		Dockerhub push time: ~ 3.5/3.9 sec
Dockerhub pull time: ~ 1.1 ses		Dockerhub pull time: ~ 0.8 ses

https://circleci.com/gh/lotharschulz/hellogodocker/91