


Advanced System Design

Zusammenfassung

Joel von Rotz /  [Quelldateien](#)

Inhaltsverzeichnis

| | |
|--|---|
| Crypto | 1 |
| Data Encryption Standard | 1 |
| AES | 1 |
| Hash Functions | 1 |
| Block | 1 |
| Docker  | 1 |
| Was'n Docka? | 2 |
| Lebenszyklus | 2 |
| Image Reference Format | 2 |
| Volume | 2 |
| Host Volumes | 2 |
| Named Volumes | 2 |
| Anonyme Volumes | 2 |
| Container → Image | 3 |
| Safety | 3 |
| Ada Spark | 3 |
| Performance | 3 |
| Cache | 3 |
| Trashing | 3 |
| Thrashing / Seitenflattern | 3 |
| Cache Struktur | 3 |
| GCC Optimization | 3 |
| | 3 |
| Trends | 3 |

STOP DOING CRYPTOGRAPHY

- DATA WAS NOT MEANT TO BE SAFE
- YEARS OF RESEARCHING yet NO ENCRYPTION FOUND more secure than ONE-TIME-PAD
- Wanted more security anyway? We have a name for that: It's called TALKING IN REAL LIFE
- “Please make sure to add SALT to your passwords”
- Statement dreamt up by the utterly deranged

LOOK at what cryptographers have been demanding your respect for all this time, with all the Computers and Abstract Algebra we made for them

(This is REAL Cryptography, done by REAL Cryptographers)



?????




Who's IKE???



Hell naw

“Hello I would like  apples Please”

They have played us for absolute fools

| | |
|---|--|
| Crypto | |
| Data Encryption Standard | |
| AES | |
| <input type="checkbox"/> Hello | |
| Hash Functions | |
| Block | |
| Docker  | |



! Wichtig

Bash Befehle via Host sind mit **\$** gekennzeichnet.

```
$ echo "this happens on WSL or SSH Pi"
```

Bash Befehle in einem Docker Container sind mit **#** gekennzeichnet.

```
# echo "this happens in a Docker container"
```

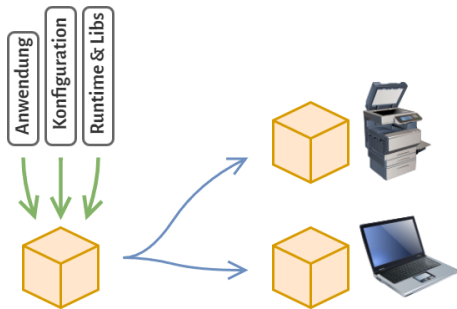
Was'n Docka?

Kurzgesagt: Docker "containerisiert" Anwendungen.

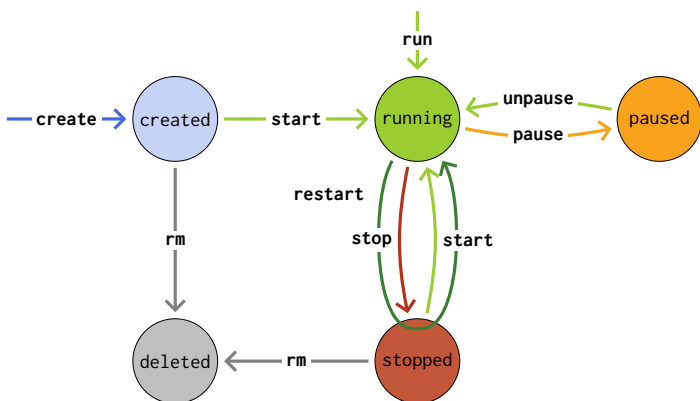
Die Idee ist, eine Anwendung mit der nötigen Konfiguration, Runtime und Bibliotheken in ein Paket zusammenzustellen und dann als **portables** Produkt weitergegeben, verarbeitet, etc. ausgeführt werden.

Man unterscheidet zwischen **Image** & **Container**

- **Image** ist eine Vorlage, welche beschreibt, wie ein Container aufzubauen ist.
- **Container** ist die ausgeführte Instanz eines Images und ist eine isolierte Umgebung, welche die entsprechenden Prozesse ausführt, **ohne** andere Systeme zu stören.

**Lebenszyklus**

Ein Docker-**Container** kann fünf Zustände annehmen (Created, Running, Deleted, Stopped und Paused) und kann mit folgenden Docker-Befehlen gesteuert werden.

**Image Reference Format**

Standardmässig werden Images wie z.B. [hello-world](#) immer vom [DockerHub-registry](#) heruntergeladen, aber es ist möglich andere **repos** anzufragen. Grundsätzlich gilt folgende Struktur:

<repo>/<source>/<image>/<tag>

- **<repo>**: Repository/Content-Host (default [index.docker.io](#))
- **<source>** Untergruppe, Hauptprojekt, User, Organisation, etc. (default [library](#))
- **<image>** Projekt, wie z.B. eine Runtime
- **<tag>** Version oder Tag des Projektes (default [latest](#))

```
$ docker run kaohslu/01-demo-img:latest
```

→ Auf dem offiziellen Repository **DockerHub** wird unter dem User **kaohslu** das Image **01-demo-img** der Version **latest** heruntergeladen und gestartet.

Volume

Da die Container voneinander isoliert sind, können diese auch nicht gegenseitig auf den Speicher zugreifen. Wenn man z.B. Dateien in einen bestimmten Ordner auf dem Host-Computer abspeichern möchte oder mehrere Container auf den gleichen Speicher zugreifen lassen, muss dies **explizit** angegeben werden.

i Volume Handling

Volumes werden durch

Host Volumes

```
$ docker run -v /path/in/host:/path/in/container
```

- **/path/in/host**: Pfad auf dem Host
- **/path/in/container**: Pfad im Container, welche mit dem Host-Pfad verbunden wird.

Named Volumes

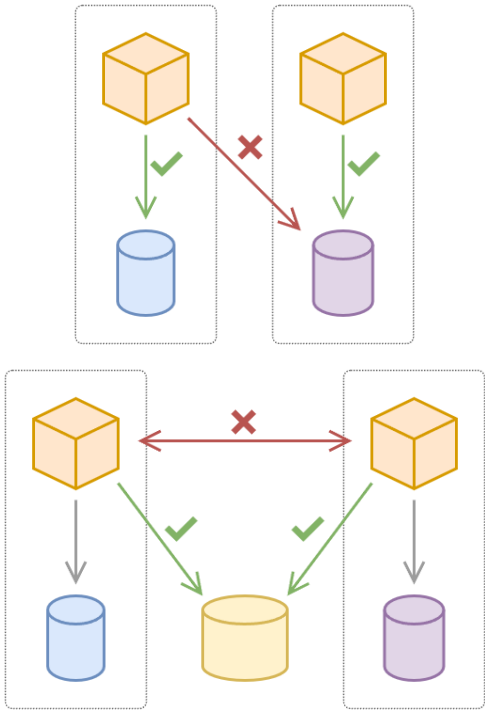
```
$ docker run -v name:/path/in/container
```

- **name**: Name des Volumens
- **/path/in/container**: Pfad im Container, welche geöffnet wird.

Anonyme Volumes

```
$ docker run -v /path/in/container
```

- **/path/in/container**: Pfad im Container, welche geöffnet wird.



💡 Enabling Optimization

#pragma GCC optimize ("O0")

Trends

Container → Image

Safety

Ada Spark

Performance

Cache

i Was Cache?

Trashing

Thrashing/Seitenflattern

Cache Struktur

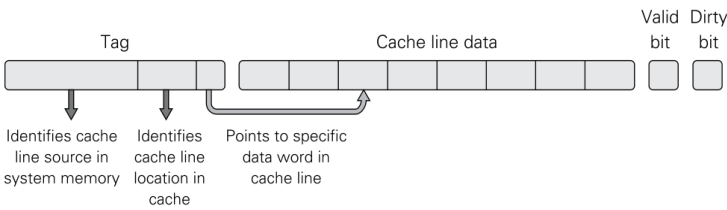


Abbildung 0.1: Struktur einer Cache-Zeilen

GCC Optimization

-O0 no optimization -O3 all optimization (-O1,-O2) + function inlining and more

funroll-loops