Dokumentation Assignment Cloud-Computing

# Infrastruktur:

|  |  |
| --- | --- |
| **Name** | **Zweck** |
| Putty[[1]](#footnote-1) | TelnetClient zur Serversteuerug und -verwaltung |
| SourceTree[[2]](#footnote-2) | Git Graphical User Interface (GUI) für eine einfachere Verwaltung des Repository |
| Git Repository auf GitHub[[3]](#footnote-3) | Bequeme Umverteilung auf den Server und auf einzelne Teammitglieder, Backupmöglichkeit, Portierung von z.B. Dockerfiles auf „private“ Testinstanzen |
|  |  |

Tabelle 1: Grundlegendes IT Setup

## Initiale Schritte

* Installation und Einrichtung aller obigen Programme
* Erstellung eines Git Repositories auf GitHub
* Installation von SourceTree
* Klonen des Repositories auf alle Teamrechner und dem Server

# Aufgaben

## Definitionen & Begriffserläuterungen

## CentOS Container

Ziel der Übung ist es,

|  |
| --- |
| $nano Dockerfile |

Code 1: Erstellen eines Dockerfiles

Der oben genannte Befehl ruft das *Programm nano* auf. Außerdem wird ein Dockerfile erstellt, welches als Grundlage für die Aufgabe dient- falls es nicht bereits vorhanden ist.

|  |
| --- |
| FROM centos:7  MAINTAINER "Florian Kaefer, Karlo Kraljic"  VOLUME /sys/fs/cgroup |

Code 2: Dockerfile für das CentOS Image

In Code 2 befindet sich der Inhalt des Dockerfile. Das FROM Schlüsselwort gibt an, welches so genannte Base Image dem Container zugrunde liegt. MAINTAINER gibt die Maintainer des Containers an, was im konkreten Fall Florian und Karlo sind.

VOLUME wird laut CentOS Homepage benötigt, um das Volume */sys/fs/cgroup* einzubinden, um ein lauffähiges Image zu gewährleisten.

|  |
| --- |
| $ docker build --no-cache -t centos:7 . |

Code 3: Starten des Build Prozesses

|  |
| --- |
| Sending build context to Docker daemon 2.56 kB  Sending build context to Docker daemon  Step 0 : FROM centos:7  7: Pulling from centos  47d44cb6f252: Pull complete  8aa7f4a1dd13: Pull complete  e0f5867add13: Pull complete  eeb3a076a0be: Pull complete  Digest: sha256:b3da5267165bbaa9a75d8ee21a11728c6fba98c0944dfa28f15c092877bb4391  Status: Downloaded newer image for centos:7  ---> eeb3a076a0be  Step 1 : MAINTAINER "Florian Kaefer, Karlo Kraljic"  ---> Running in d1f76749f30b  ---> 890de4e7421c  Removing intermediate container d1f76749f30b  Step 2 : VOLUME /sys/fs/cgroup  ---> Running in e4a2512db239  ---> 368f6b056e72  Removing intermediate container e4a2512db239  Step 3 : CMD /bin/bash  ---> Running in 8ee4493c13a8  ---> 7892c2457995  Removing intermediate container 8ee4493c13a8  Successfully built 7892c2457995 |

Code 4: Bash Output nach erfolgreichem Build Prozess

|  |
| --- |
| $docker run -t -i centos:7 ./bin/bash |

Code 5: Starten der Bash im Container

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Bash Kommando** | **Außerhalb des Container** | **Innerhalb des Containers** |
| Laufende Prozesse  (Prozess ID + Prozess) | ps -A | 1 init      2 kthreadd      3 ksoftirqd/0      5 kworker/0:0H      7 rcu\_sched      8 rcu\_bh      9 rcuos/0     10 rcuob/0     11 migration/0     12 watchdog/0     13 khelper     14 kdevtmpfs     15 netns     16 perf     17 khungtaskd  […] | 1 bash  17 ps |
| Distribution | cat /etc/\*-release | Ubuntu 14.04.4 LTS, Trusty Tahr | CentOS Linux 7 |
| Kernel Version | uname -r | 3.19.0-49-generic | 3.19.0-49-generic |

## Grundlegende Containerbefehle

|  |  |
| --- | --- |
| Docker Befehl | Beschreibung |
| docker ps <flags> | Dieser Befehl gibt eine Liste mit den Laufenden Containern aus. Es können optional verschiedene Flags angehängt werden- der –l Flag gibt zum Beispiel den aktuellsten Container (unabhängig vom Ausführungsstatus, d.h. Running-Exited oder Paused) aus. Das –a Flag gibt alle verfügbaren Container aus. |
| docker info <cont.ID> |  |
| docker top <cont. ID> |  |
| docker stop <cont. ID> |  |

## Docker Hub

1. http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html [↑](#footnote-ref-1)
2. https://www.sourcetreeapp.com/download/ [↑](#footnote-ref-2)
3. https://github.com/KKraljic/cloud-computing-assignments/ [↑](#footnote-ref-3)