W901 - Grundlagen Linux, UNIX Betriebsystem

E010 - LINUX ESSENTIALS EXAM 010

Murugathas Kavinth

TBZ

Inhaltsverzeichnis

Fahrplan ***2***

1.1 Linux Evolution and Popular Operating Systems (Status: Erledigt) **3**

1.3 Open Source Software and Licensing **4**

2.1 Command Line Basics **5**

2.3 Using Directories and Listing Files **6**

2.4 Creating, Moving and Deleting Files **7**

3. 1 Archiving Files on the Command Line **8**

3.2 Searching and Extracting Data from Files **9**

3.3 Turning Commands into a Script **10**

4.2 Understanding Computer Hardware **11**

4.4 Your Computer on the Network **12**

702.1 Container Usage – Studium **14**

702.1 Container Usage – Umsetzung **17**

Fahrplan

Datum	behandelte Unterrichtsinhalte:	Gewichtung
15.05.19	1.1 Linux Evolution and Popular Operating Systems 1.3 Open Source Software and Licensing	2 + 2
22.05.19	2.1 Command Line Basics 2.3 Using Directories and Listing Files 2.4 Creating, Moving and Deleting Files	2 + 3 + 2
29.05.19	3.2 Searching and Extracting Data from Files 3.3 Turning Commands into a Script	3 + 4
05.06.19	4.4 Your Computer on the Network	2
12.06.19	702.1 Container Usage - Studium	-
19.06.19	702.1 Container Usage - Umsetzung	7 (14)
26.06.19	LB1 Theoretische Prüfung	-
03.07.19	Lehrer Gesamtkonferenz	-
10.07.19	Abgabe LB2	-
	Total Punkte	24 (31)

Kapitel aus E010 sind einzeln erarbeitet worden.

1.1 Linux Evolution and Popular Operating Systems (Status: Erledigt)

Weight: 2

Beschreibung:
Linux Geschichte und weitere Betriebssysteme

Tagesziele:

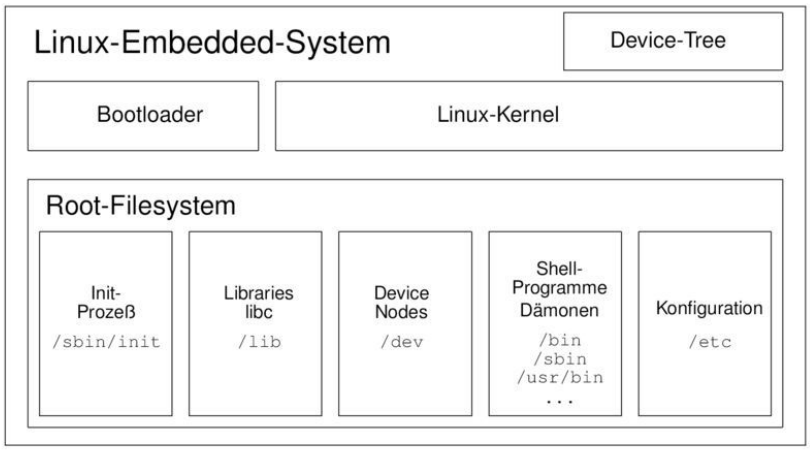
1. Installation SW, Einrichten Linux VM
2. Theorie einlesen (Linux Essentials von tuxcademy - Kapitel 1)

Vorgehen: Theorie erarbeiten und danach VM aufsetzen

Beispiele und Arbeitsergebnisse:

Distributionen:
Linux-Distributionen erweitern den Linux-Betriebssystemkern mit Benutzerprogrammen und Dokumentation zu einem tatsächlich benutzbaren
System.

Embedded Systems:
Ein embedded System ist ein Teil eines noch grösseren Systems, das in der Regel eine wesentliche Funktion für den Betrieb des größeren Geräts übernimmt.
Als Embedded Linux bezeichnet man ein eingebettetes System mit einem auf dem Linux-Kernel basierenden Betriebssystem



Cloud Computing
Die Flexibilität von Linux macht es auch zum Betriebssystem der Wahl für Anwendungen wie Cloud Computing. Cloud Computing ist im Grunde die Bereitstellung von Computingressourcen (z.B. Server, Speicher, Datenbanken, Netzwerkkomponenten, Software, Analyse- und intelligente Funktionen) über das Internet, um schnellere Innovationen, flexible Ressourcen und Skaleneffekte zu bieten.

1.3 Open Source Software and Licensing

Weight: 2

Beschreibung:
Offene Communities und Lizenzierung von Open Source Software für Unternehmen.

Tagesziele: siehe 1.1

Vorgehen: Mit den gegebenen Unterlagen die nötigen Informationen erarbeiten.

Beispiele und Arbeitsergebnisse:

Urheberrecht
Das Urheberrecht beschreibt nicht nur den offiziell anerkannten Anspruch auf das persönliche Eigentum, sondern legt auch fest, wer dieses Eigentum in welchem Rahmen nutzen darf.

Freie Software
Der Benutzer hat die Erlaubnis, verschiedene Dinge zu tun, die er mit nicht frei zugängliche Software nicht könnte oder dürfte.

Open-Source Software
Open Source bedeutet, dass der Quellcode einer Software offen und frei zugänglich ist.
So ist es gestattet das Programm selbst bearbeiten zu können.
Die geänderte und verbesserte Programm-Datei dürfen auch von Drittpersonen auch weiterverbreitet werden.

Lizenzen
Die Lizenzen für freie und Open-Source-Software dagegen dienen dazu,
dem Erwerber der Software Dinge zu erlauben, die er laut Urheberrechtsgesetz eigentlich sonst nicht dürfte
Copyleft-Lizenzen: Software kann in seine eigenen Programme integriert und diese Programme in ausführbarer Form weitergeben werden.

Fazit und Aussicht (Tag 1)

Das Aufsetzen des Ubuntu VM's lief gut, hatte keine Schwierigkeiten dabei. Ich habe heute etwas über die Geschichte von Linux gelernt und woraus ein Linux-System besteht (Bootloader, Linux-Kernel und Root-Filesystem) und ein Embedded-Linux-System aufgebaut ist. Ausserdem habe ich etwas zu Neues zu Lizenzen gelernt und finde, ist ein sehr heikles Thema, da man recht gut aufpassen muss, was man macht. Nächste Woche schaue ich mir das Terminal genauer an.

2.1 Command Line Basics

Weight: 2

Beschreibung:
Grundlagen der Verwendung der Linux-Befehlszeile.

Tagesziele:

1. Grundbefehle kennenlernen
2. Files bearbeiten
3. Wissensaufbau im Filesystem Linux

Vorgehen: Die Befehle kennenlernen und gleich anwenden

Beispiele und Arbeitsergebnisse:
Standard für Linux-Systeme ist die Bourne-Again-Shell, kurz **bash**. Kommandos in dieser Sprache müssen gewissen Regeln, einer **Syntax**, gehorchen, damit sie von der Shell verstanden werden können.
Variablen sind symbolische Namen für Werte und verleihen einem Skript große Flexibilität. Sie erlauben es, einen Wert an nur einer Stelle zu ändern und den Wert überhaupt zu ändern.
Das **Quoting** dient dazu, bestimmte Zeichen mit einer Sonderbedeutung vor der Shell zu 'verstecken' um zu verhindern, dass diese expandiert (ersetzt) werden.

Die allgemeine Befehlsstruktur:

- Kommando – Was wird gemacht?
- Optionen – Wie wird es gemacht?
- Argumente – Womit wird es gemacht?

echo	Kommando, das einfach nur seine Parameter ausgibt
history	Durch Eingabe von history lässt sich auch eine Liste der eingegebenen Befehle ausgeben
date	Gibt Datum und Uhrzeit aus
export	Das Kommando export sorgt dafür, dass die Variable nicht nur in der aktuellen Shell, sondern auch in den von der aktuellen Shell aus aufgerufenen Programmen zur Verfügung steht
type	Bestimmt die Art eines Kommandos (intern, extern) **Interne Kommandos:** Sind von der Shell selber und ihr Vorteil ist, dass sie schnell ausgeführt werden können. Sie beeinflussen den Zustand der Shell selbst **Externe Kommandos:** Kommando startet eine ausführbare Datei.
help	Zeigt Hilfe für bash-Kommandos

2.3 Using Directories and Listing Files

Weight: 3

Beschreibung: Navigation von Home- und Systemverzeichnissen und Auflistung von Dateien an verschiedenen Orten.

Tagesziele:
Siehe 2.1

Vorgehen: Die Befehle kennenlernen und gleich anwenden

Beispiele und Arbeitsergebnisse:
Verzeichnisse dienen zur Strukturierung des Speicherplatzes.
Das Filesystem in Linux basiert auf das Rootverzeichnis (/). Für einen normalen User ist das Home-Verzeichnis ein wichtiges Thema (abgeküzt mit ~). Innerhalb dieses Verzeichnisses kann der Benutzer beliebige weitere Verzeichnisse anlegen und/oder Dateien speichern.
Versteckte Dateien in Linux sind Konfigurations-Dateien und Verzeichnisse, sie beginnen stets mit einem «.» (Punkt) vor der Dateiname. Um sie anzuzeigen muss der Begfehl ls -a ausgeführt werden

mount	
ls	Dateien und Verzeichnisse auflisten
cd	Ins Verzeichnis wechseln

Hier sind ein paar Verzeichnisse aufgelistet, welche man kennen sollte:

/	Die erste Verzeichnisebene.
/bin	Muss vorhanden sein, da in diesem Verzeichnis wichtige Programme für die User befinden
/dev	Dieses Verzeichnis enthält alle wichtige Gerätedateien, über der die Hardware im Betrieb angesprochen werden
/etc	Es sind alle Konfgurations- und Informationsdateien
/home	Ort, an dem Benutzer ihre Daten ablegen können und an dem Programme ihre benutzerspezifischen Einstellungen hinterlegen.
/proc	Enthält Schnittstellen zum Kernel und seine Prozesse
/root	Homeverzeichnis des Superusers. Das Homeverzeichnis des Users ist nicht in /home, weil das Verzeinis des Roots immer erreichbar sein muss

/tmp	Im Verzeichnis /tmp werden von einigen Programmen temporäre Dateien zur Zwischenspeicherung von Laufzeitdaten angelegt.
/usr	user enthält die meisten Systemtools, Bibliotheken und installierten Programme

2.4 Creating, Moving and Deleting Files

Weight: 2

Beschreibung:
Erstellen, verschieben und löschen von Dateien und Verzeichnisse unter dem Home-Verzeichnis.

Tagesziele:
Siehe 2.1

Vorgehen: Die Befehle kennenlernen und gleich anwenden

Beispiele und Arbeitsergebnisse:

cp	Mit dem Kommando cp kann man beliebige Daten kopieren
mv	Damit kann man eine Datei an einen anderen Ort abspeichern oder den Namen verändern
rm	Datei löschen (Schreibrecht muss vorhanden sein)
ln	Erzeugt eine Verknüpfung zu einer Datei oder einem Verzeichnis.
mkdir	Verzeichnisse anlegen
rmdir	Verzeichnisse löschen
touch	leeres File erstellen

Fazit und Aussicht (Tag 2):
Mit Linux habe ich bereits in früheren Modul gearbeitet, doch ich kannte mich da nichts so gut aus. Die Grundbefehle durchzugehen hat mir wirklich sehr geholfen und ich fühle mich schon sicherer mit der Shell zu arbeiten. Das Quoting war für mich ein schwieriges Theam, da ähnlich aussehende Zeichen völlig verschiedene Effekte bewirken und ich nicht ganz verstanden habe, wieso man es einsetzt. Ich verstehe nun auch das Filsystem von Linux mehr und kann auch darin navigieren. Wie man mit Files arbeitet, habe ich früher auch schon gelernt gehabt. Der Begriff «simple globbing» ist mir fremd, was ich noch klären werde und nächste Woche werde ich mich genauer auf das Kapitel 3 (Fahrplan) fokussieren.

3. 1 Archiving Files on the Command Line

Weight: 2

Description: Archivieren von Dateien im Home-Verzeichnis des Benutzers

Beispiele und Arbeitsergebnisse:

Archivierung: Prozess des Zusammenfassens vieler Dateien zu einer einzigen.
Komprimierung: Umschreiben von Daten in eine gegenüber dem Original platzsparende Fassung. Es geht um die »verlustfreie« Komprimierung, bei der es möglich ist, aus den komprimierten Daten das Original in identischer Form wieder herzustellen.

tar
Einzelne Dateien werden hintereinander in eine Archivdatei gepackt und mit Zusatzinformationen (Zugriffsrechte, Eigentümer etc.) versehen.

Kommando: *tar ⟨Optionen⟩ ⟨Datei⟩| ⟨Verzeichnis⟩*

Die wichtigsten Optionen sind:

- -c erzeugt ein neues Archiv
- -f «Datei» erzeugt oder liest das Archiv von «Datei», wobei dies eine Datei oder ein Gerät sein kann
- -t zeigt den Inhalt des Archivs
- -x Auslesen der gesicherten Dateien
- -M bearbeitet ein tar-Archiv, das sich über mehrere Datenträger erstreckt

gzip
Dient dazu, einzelne Dateien zu komprimieren oder auch Archive, die viele Dateien enthalten.

bzip2
bzip2 verwendet ein anderes Verfahren als gzip, das zu einer höheren Komprimierung führt, aber mehr Zeit und Speicherplatz zum Komprimieren benötigt.

xz
xz bietet noch bessere Komprimierung als bzip2, allerdings bezahlt man mit der wiederum längeren Komprimierungszeiten (sowie Entkomprimierung). xz bietet sich vor allem für die Software-Verteilung an, wo Quellcodes oder Binärpakete typischerweise einmal zusammengepackt, aber oft übers Internet übertragen und wieder ausgepackt werden

3.2 Searching and Extracting Data from Files

Weight: 3

Beschreibung: Suchen und extrahieren von Daten aus Dateien im Home-Verzeichnis.

Tagesziele:

- Einfaches Bash Script in Linux erstellen
- I/O redirection anwenden

Vorgehen: Theorie erarbeiten und in der Praxis anwenden

Beispiele und Arbeitsergebnisse:

Kommando Pipelines

Linux bietet eine direkte Verknüpfung von Kommandos durch Pipes an. Damit wird die Ausgabe eines Programms automatisch zur Eingabe des nächsten Programms

I/O redirection

Es werden zwei verschiedene Kanäle eingesetzt für die verschiedene Datenströme.
Kanal 1 = Standard Ausgabe (stdout)
Kanal 2 = Standard Fehlerkanal (stderr)

stdin:
Über die Standardeingabe können Daten in ein Programm eingelesen werden
(Bsp. Tastatur).

stdout:
Über die Standardausgabe kann ein Programm Daten ausgeben (Bsp. Monitor).

stderr:
Die Standardfehlerausgabe ist ein zweiter Ausgabedatenstrom, der dazu gedacht ist, Fehler- und Statusmeldungen auszugeben.

grep	Sucht in Dateien nach Zeilen mit bestimmtem Inhalt
less	Eine bestimmte Anzahl Zeilen einer Datei anzeigen
cat	Fileinhalt ausgeben
head	Gibt die ersten Zeilen einer Datei aus
tail	Gibt die letzten Zeilen einer Datei aus
sort	Sortiert die Ausgabe von Programmen, bzw. Dateien
cut	Gibt bestimmte Teile eines Textes aus

3.3 Turning Commands into a Script

Weight: 4

Beschreibung: Befehle in einfache Skripte verwandeln.

Tagesziele: siehe 3.2

Vorgehen: Durcharbeiten Dokumentation Kapitel 9

Beispiele und Arbeitsergebnisse:

Linux unterstützt beliebige Scriptsprachen und der erste Eintrag in einer Datei bestimmt die Script-Umgebung. Der Hashbang (**#!** (shebang)) steht am Anfang eines Scriptes.
Beispiel:
#!/bin/bash – Bash Script
#!/usr/bin/node – Node JavaScript
#!/usr/bin/perl – Perl

Argumente dienen nicht zur Steuerung eines Kommandos, sondern liefern diesem Informationen, die es zu bearbeiten hat. Viele Kommandos zur Manipulation von Dateien benötigen zum Beispiel die Namen der Dateien, die sie manipulieren sollen. Hier wird also nicht das Verhalten des Programmes geändert, sondern die Information variiert, die dem Programm für seine Arbeit zur Verfügung steht.

For loop
Ein «for-loop» ist eine bash-Anweisung, die es ermöglicht, Code wiederholt auszuführen.
Ein Beispiel:

Dies ist ein for-loop, welches durch Zählen gekennzeichnet ist.
Der Bereich wird durch eine Anfangs- (#1) und Endnummer (#5) angegeben. Die for-loop führt eine Folge von Befehlen für jedes Element in einer Liste von Elementen aus. Ein repräsentatives Beispiel in BASH ist wie folgt, um die Begrüßungsnachricht 5 mal mit for loop anzuzeigen:

```
#!/bin/bash
for i in 1 2 3 4 5
do
   echo "Welcome $i times"
done
```

Mit dem Befehl **echo** lassen sich Zeichenketten und Variablen zeilenweise auf dem Standardausgabegerät anzeigen.

```
echo 'Das ist ein Beispiel!'
```

```
Das ist ein Beispiel!
```

Befehlsoptionen für echo:

\\	Ausgabe des Backslashs
\b	Backspace
\n	Neue Zeile
\e	Escape

Fazit und Aussicht (Tag 3):
Heute habe ich den Grundsatz von Linux kennengelernt und damit gearbeitet und zudem auch weitere sehr wichtig Befehle zum Suchen und extrahieren von Daten angeschaut und umgesetzt. Ich habe weitere Grundlagen zu Arbeiten mit Terminal erlernt.

4.2 Understanding Computer Hardware

Weight: 1

Motherboard
Hauptplatine des Computers, auf der alle wesentlichen Bauteile angeordnet sind

Prozessor
Der Prozessor (CPU) ist das Herzstück des Computers. Hier findet die automatische programmgesteuerte Datenverarbeitung statt, die den Computer erst zum Computer macht.

Peripheriegeräte
Als ein Peripheriegerät bezeichnet man eine Komponente oder ein Gerät das man auch als Zubehör eines Computers bezeichnen kann.

Festplatte
Eine Festplatte ist das Speichermedium in einem Computer. Alle Dokumente, Bilder oder Programme werden dort gesichert.

Treiber
Die Aufgabe von Treibern ist es, am Computer angeschlossene Hardware zu erkennen und mit dieser zu kommunizieren, so genannte Vermittler. Treiber übersetzen die Sprache der jeweiligen Hardware (Drucker) für den Computer.

4.4 Your Computer on the Network

Weight: 2

Beschreibung:
Wichtigste Netzwerkkonfiguration und Bestimmung der grundlegenden Anforderungen an einen Computer in einem lokalen Netzwerk (LAN) abfragen.

Tagesziele:

- Client «netzfähig machen»
- Commands kennenlernen, welche im Netzwerkbereich verwendet werden

Vorgehen: Theorie lesen und die Befehle kennenlernen

Beispiele und Arbeitsergebnisse:
Anforderungen damit ein Linux Rechner als Client in ein existierendes Netz integriert werden kann:

- IP-Adresse für den Rechner
- Netzwerkadresse und -maske für den Rechner
- Adresse eines Defaults-Gateways im lokalen Netz
- Adresse eines DNS-Servers

route:
Mit diesem Befehl kann die Routing-Tabelle abgerufen werden.

```
kavinth@modul901:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         _gateway        0.0.0.0         UG    100    0        0 enp0s3
10.0.2.0        0.0.0.0         255.255.255.0   U     0      0        0 enp0s3
_gateway        0.0.0.0         255.255.255.255 UH    100    0        0 enp0s3
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0        0 docker0
```

ping
Mit dem Kommando ping kann man auf der untersten Ebene (IP) prüfen, ob die Maschine mit anderen Computern kommunizieren kann. ping benutzt das Steuerungsprotokoll ICMP, um einen anderen Rechner um ein Signal zu bitten.

netstat
Mit dem Kommando netstat kann man Informationen aller Art über seinem Rechner und seine Netzanbindung herausfinden
netstat -tl: Mit netstat -tl kann man eine Liste der TCP-Ports ausgeben lassen, auf dem lokalen Rechner Serverprogramme auf eingehende Verbindungen lauschen.

```
kavinth@modul901:~$ netstat -tl
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 localhost.loca:http-alt 0.0.0.0:*               LISTEN
tcp        0      0 localhost.localdo:10256 0.0.0.0:*               LISTEN
tcp        0      0 localhost:domain        0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN
tcp        0      0 localhost.localdom:1338 0.0.0.0:*               LISTEN
tcp        0      0 localhost.localdo:33629 0.0.0.0:*               LISTEN
tcp        0      0 localhost.localdo:10248 0.0.0.0:*               LISTEN
tcp        0      0 localhost.localdo:10249 0.0.0.0:*               LISTEN
tcp        0      0 localhost.localdo:39723 0.0.0.0:*               LISTEN
tcp        0      0 localhost.localdo:10251 0.0.0.0:*               LISTEN
tcp        0      0 localhost.localdo:10252 0.0.0.0:*               LISTEN
tcp        0      0 localhost.localdom:2380 0.0.0.0:*               LISTEN
tcp6       0      0 [::]:10255              [::]:*                  LISTEN
tcp6       0      0 [::]:10257              [::]:*                  LISTEN
tcp6       0      0 [::]:10259              [::]:*                  LISTEN
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN
tcp6       0      0 [::]:16443              [::]:*                  LISTEN
tcp6       0      0 [::]:10250              [::]:*                  LISTEN
tcp6       0      0 [::]:35531              [::]:*                  LISTEN
```

/etc/resolv.conf

Die Datei resolv.conf im Verzeichnis /etc wird für die Namensauflösung entsprechend dem | Domain Name System (DNS) verwendet

/etc/hosts

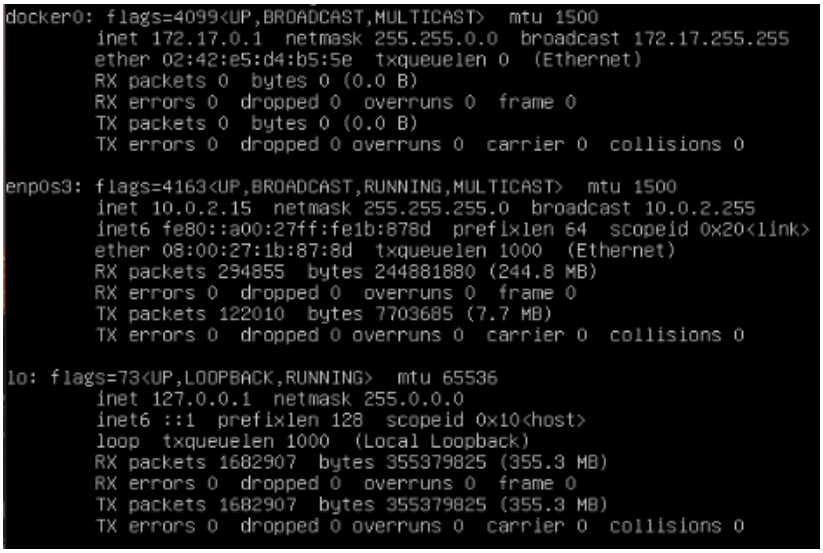
Die Datei /etc/hosts wird benutzt, um Rechnernamen in IP-Adressen aufzulösen, wenn kein Nameserver im lokalen Netzwerk vorhanden ist.

IPv6

Statt 32 Bit breiten Adressen verwendet IPv6 128 Bit breite Adressen, damit es in der Zukunft zu keinen Schwierigkeiten kommt.

ifconfig

Das Kommando ifconfig wird benutzt, um Netzwerkschnittstellen zu konfigurieren. Es wird meistens beim Booten des Systems verwendet, um die Schnittstellen zu konfigurieren. Wenn keine Argumente angegeben werden, dann zeigt ifconfig den Zustand der augenblicklich aktiven Netzwerkschnittstellen



Fazit und Aussicht (Tag 4):

Ich habe die Netzwerkeinstellungen in Linux kennengelernt. Nun bin ich sicherer im Linux Umfeld und habe das nötige Wissen das System zu nutzen. Nächste Woche schaue ich mir Docker an.

702.1 Container Usage – Studium

Microservices

Hinter Microservices stecken ebenfalls Container, daher hatte man damit auch so ein Erfolg. Microservices sind ein Ansatz zur Modularisierung des Softwares. Sie nutzen als Module einzelne Programme, die als eigene Prozesse laufen.
Die 3 Aspekte der Microservice:

- Ein Programm soll nur eine Aufgabe erledigen, und das soll es gut machen.
- Programme sollen zusammenarbeiten können.
- Nutze eine universelle Schnittstelle. In UNIX sind das Textströme bei Microservices das Internet.

Pod

Eine Gruppe aus einem oder mehreren Containern, die in einem einzelnen Knoten implementiert wurde. Alle Container in einem Pod teilen sich die IP-Adresse, IPC, Hostname und andere Ressourcen.

Docker

Docker nahm die bestehende Linux-Containertechnologie auf und verpackte und erweiterte sie in vielerlei Hinsicht vor allem durch portable Images und eine benutzerfreundliche Schnittstelle, um eine vollständige Lösung für das Erstellen und Verteilen von Containern zu schaffen.

Docker Architektur

Docker Daemon

Der Docker-Daemon (dockerd) wartet auf Docker-API-Anfragen und verwaltet Docker-Objekte wie Images, Container, Netzwerke und Volumes.

Docker Client

Der Docker-Client (Docker) ist die primäre Methode, mit der viele Docker-Benutzer mit Docker interagieren. Wenn man Befehle eingibt, sendet der Client diese Befehle an dockerd, welche sie ausführt. Der Docker-Befehl verwendet die Docker-API.

Images

Images sind gebuildete Umgebungen welche als Container gestartet werden können und sind nicht veränderbar

Container

Container sind die ausgeführten Images und ein Image kann beliebig oft als Container ausgeführt werden.

Merkmale

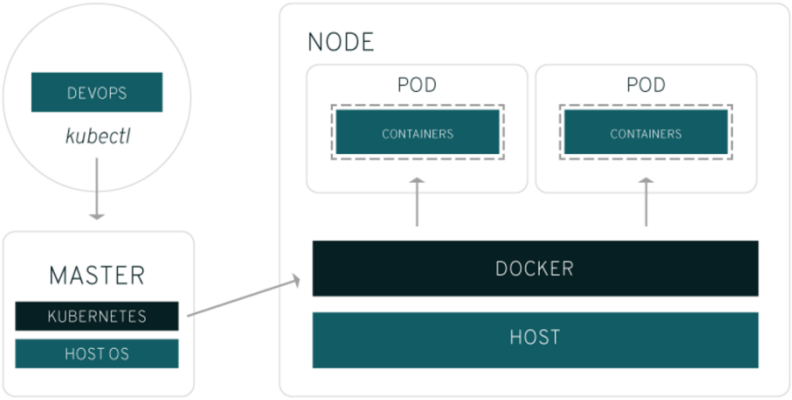
- Container teilen sich Ressourcen mit dem Host-Betriebssystem
- Container können im Bruchteil einer Sekunde gestartet und gestoppt werden
- Anwendungen, die in Containern laufen, verursachen wenig bis gar keinen Overhead
- Container sind leichtgewichtig, d.h. es können dutzende parallel betrieben werden.
- Container sind "Cloud-ready"!

Dockerfile

Darin wird der Aufbau eines Docker Images beschrieben

Kubernetes

Kubernetes ist eine Open Source-Plattform, die den Betrieb von Linux-Container automatisiert.
Kubernetes läuft aufgesetzt auf ein und interagiert mit den Pods von Containern, die auf den Knoten laufen. Der Kubernetes Master empfängt Befehle vom Administrator und leitet diese Befehle an die untergeordneten Knoten weiter. Diese Übergabe arbeitet mit einer Vielzahl von Services, die automatisch entscheiden, welcher Knoten am besten für die Aufgabe geeignet ist. Der Knoten weist dann Ressourcen zu und legt die Pods in diesem Knoten fest, die die gewünschte Aufgabe durchführen sollen.



702.1 Container Usage – Umsetzung

Weight: 7 (14)

Beschreibung: Arbeiten mit Container

Tagesziele:

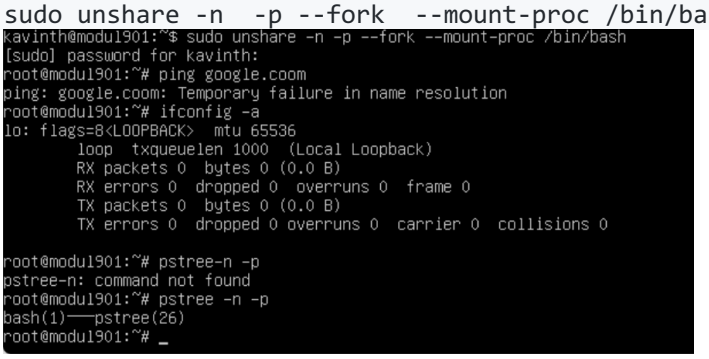
1. Erste Schritte mit Namespace
2. Erste Schritte mit Docker

Vorgehen: Modul 300 Unterlagen verstehen und anwenden.

Beispiele und Arbeitsergebnisse:

Heute habe ich mit Namespace gearbeitet gehabt, die Linux Variante von einem Container. Dazu muss auf der Linux Maschine dieses Befehl ausgeführt werden:

```
sudo unshare -n  -p --fork  --mount-proc /bin/bash
```



Ausserdem weiss ich nun wie ich mithilfe Docker, Containers starten kann. Dazu muss sie zuerst heruntergeladen werden:

```
docker run -it ubuntu /bin/bash
```

docker run	Befehl zum Starten neuer Container. Man kann damit konfigurieren, wie das Image laufen soll, Dockerfile Einstellungen überschreiben.
docker ps	Überblick über die aktuellen Container (Namen, IDs und Status)
docker images	Gibt eine Liste lokaler Images aus, wobei Informationen zu Repository-Namen, Tag-Namen und Grösse enthalten sind.
docker rm	Entfernt einen oder mehrere Container. Gibt die Namen oder IDs erfolgreich gelöschter Container zurück.
docker rmi	Löscht das oder die angegebenen Images. Diese werden durch die ID oder Repository- und Tag-Namen spezifiziert.
docker start	Startet einen gestoppten Container

Fazit und Aussicht (Tag 6):

Ein richtiges Projekt mit Containern habe ich nicht gemacht, da mir die Erfahrung fehlte, ich hatte das Modul 300 leider nicht, und der Zeitraum zu knapp wurde. Doch ich habe in diesem Modul eine solide Grundlage aufbauen können und gesehen was alles möglich ist durch das Containisieren und werde in naher Zukunft damit arbeiten.