

DOKUMENTATION W901

Philipp Zahner

Inhaltsverzeichnis

1. Fahrplan Wahlmodul 901	4
2. Lernjournal	5
17.05.2019 - Tag 1.....	5
22.05.2019 – Tag 2.....	5
29.05.2019 – Tag 3.....	5
05.06.2019 – Tag 4.....	5
12.06.2019 – Tag 5.....	5
19.06.2019 – Tag 6.....	6
3. E010 - Linux Essentials Exam 010.....	7
3.1 Topic 1: The Linux Community and a Career in Open Source.....	7
3.1.1 Linux Evolution and Popular Operating Systems	7
3.1.2 Major Open Source Applications.....	10
3.1.3 Open Source Software and Licensing	14
3.1.4 ICT Skills and Working in Linux.....	18
3.2 Topic 2: Finding Your Way on a Linux System	20
3.2.1 Command Line Basics	20
3.2.1.1 Basic Shell and Command Line Syntax.....	21
3.2.1.2 Variabeln.....	22
3.2.1.3 Quoting	23
3.2.2 Using the Command Line to Get Help	24
3.2.2.1 Man Pages.....	24
3.2.2.2 Info Pages.....	25
3.2.3 Using Directories and Listing Files.....	26
3.2.3.1 Files, directories, Hidden Files and Hidden directories	27
3.2.3.2 Absolute and relative paths	28
3.2.4 Creating, Moving and Deleting Files.....	29
3.2.4.1 mkdir - Create new Folders.....	30
3.2.4.2 touch – Create files.....	31
3.2.4.3 mv – Move and Rename Files	31
3.2.4.4 Delete Files and Directories	32
3.3.1 Topic 3: The Power of the Command Line.....	33
3.3.1.1 Archiving Files and Directories	34
3.3.1.2 Different Compression Methods	36
3.3.2 Searching and Extracting Data from Files.....	37
3.3.3 Turning Commands into a Script.....	39

3.3.3.1 #! (shebang)	40
3.3.3.2 Variables	40
3.3.3.3 Argumente	40
3.3.3.4 for-Loops	41
3.3.3.5 Echo	41
3.3.3.6 Exit and Exit Status	42
3.4 Topic 4: The Linux Operating System	43
3.4.1 Choosing an Operating System	43
3.4.1.1 Differences between Windows, OS X and Linux	43
3.4.1.2 Distribution life cycle management.....	44
3.4.2 Understanding Computer Hardware	45
3.4.2.1 Aus was besteht ein Computer?.....	46
3.4.3 Where Data is Stored	48
3.4.3.1 Where are Programs and Configurations stored ?.....	49
3.4.3.1.1 / – The Root Directory	50
3.4.3.1.2 /bin – Essential User Binaries.....	51
3.4.3.1.3 /boot – Static Boot Files.....	52
3.4.3.1.4 /cdrom – Historical Mount Point for CD-ROMs.....	52
3.4.3.1.5 /dev – Device Files	52
3.4.3.1.6 /etc – Configuration Files	53
3.4.3.1.7 /home – Home Folders	54
3.4.3.1.8 /lib – Essential Shared Libraries.....	54
3.4.3.1.9 /lost+found – Recovered Files.....	54
3.4.3.1.10 /media – Removable Media	54
3.4.3.1.11 /mnt – Temporary Mount Points.....	54
3.4.3.1.12 /opt – Optional Packages	55
3.4.3.1.13 /proc – Kernel & Process Files	55
3.4.3.1.14 /root – Root Home Directory	55
3.4.3.1.15 /run – Application State Files	55
3.4.3.1.16 /sbin – System Administration Binaries	56
3.4.3.1.17 /selinux – SELinux Virtual File System.....	56
3.4.3.1.18 /srv – Service Data	56
3.4.3.1.19 /tmp – Temporary Files.....	56
3.4.3.1.20 /usr – User Binaries & Read-Only Data	57
3.4.3.1.21 /var – Variable Data Files	57
3.4.4 Your Computer on the Network.....	58

4. Kubernetes Installieren	59
4.1 Pre-Installation Steps On Both Master & Slave (To Install Kubernetes).....	59
4.2 Turn Off Swap Space	59
4.3 Update The Hosts File With IPs Of Master & Node.....	60
4.4 Setting Static IP Addresses.....	61
4.5.1 Install OpenSSH-Server	61
4.5.2 Install Docker	61
4.5.3 Install kubeadm, Kubelet And Kubectl	62
4.5.4 Updating Kubernetes Configuration.....	62
4.6 Steps Only For Kubernetes Master VM (kmaster)	63
4.7 Steps For Only Kubernetes Node VM (knode)	68
5 Hinzufügen eines Nginx Servers in Kubernetes	69

1. Fahrplan Wahlmodul 901

Datum	Behandelte Unterrichtsinhalte:	Gewichtung
15.05.19	Installation SW, Einrichten Linux VMs, Erstellung des Fahrplans, Dokumentation vorbereiten	
22.05.19	1.1 Linux Evolution and Popular Operating Systems 1.2 Major Open Source Applications 1.3 Open Source Software and Licensing 1.4 ICT Skills and Working in Linux	2+2+1+2 = 7
29.05.19	2.1 Command Line Basics 2.2 Using the Command Line to Get Help 2.3 Using Directories and Listing Files 2.4 Creating, Moving and Deleting Files	3+2+2+2 = 9
05.06.19	3.1 Archiving Files on the Command Line 3.2 Searching and Extracting Data from Files 3.3 Turning Commands into a Script	2+3+4 = 9
12.06.19	4.1 Choosing an Operating System 4.2 Understanding Computer Hardware 4.3 Where Data is Stored 4.4 Your Computer on the Network	1+2+3+2 = 8
19.06.19	5.1 Basic Security and Identifying User Types 5.2 Creating Users and Groups 5.3 Managing File Permissions and Ownership 5.4 Special Directories and Files	2+2+2+1 = 7
26.06.19	702.1 Container Usage	

2. Lernjournal

17.05.2019 - Tag 1

- **VM Aufgesetzt**
 - ComputerName: ApachePro
 - Username: user
 - PW: Asdf1234
- **Fahrplan erstellt**

22.05.2019 – Tag 2

- **Intro Herr Bernet;**
702.1 Container Usage, Change root, Unshare, namespaces, Docker, iptable, Cubernetes
Vorführung mit 20 Apache Instanzen, www.cncf.io,
Folie: <https://github.com/w901-fr19-mi/E701#7021-container-usage>
- **1.1 Linux Evolution and Popular Operating Systems**
- **1.2 Major Open Source Applications**
- **1.3 Open Source Software and Licensing**
- **1.4 ICT Skills and Working in Linux**

29.05.2019 – Tag 3

- Intro Herr Bernet ; 701.2 Standard Components and Platforms for Software,

05.06.2019 – Tag 4

- Intro Herr Bernet; Synchrone und Asynchrone Microservices,
- Weiterarbeit Kubernetes Installation

12.06.2019 – Tag 5

- Intro Herr Bernet; 701.3 Source Code Management, 701.4 Continuous Integration and Continuous Delivery
- Kubernetes fertig aufgesetzt und Nginx als Service installiert

19.06.2019 – Tag 6

- Besprechung für anstehende Prüfung am 26.06.2019 von Herr Bernet
- Weiterarbeit an Dokumentation

26.06.2019 – Tag 7

- Fertigstellung Dokumentation

3. E010 - Linux Essentials Exam 010



3.1 Topic 1: The Linux Community and a Career in Open Source

3.1.1 Linux Evolution and Popular Operating Systems

Beschreibung: Kenntnisse in der Linux-Entwicklung und wichtigen Distributionen

Wichtige Wissensgebiete:

- Distributionen
- Embedded Systems
- Linux in der Cloud

Im Folgenden finden Sie eine unvollständige Liste der verwendeten Dateien, Begriffe und Dienstprogramme:

- Debian, Ubuntu (LTS)
- CentOS, openSUSE, Red Hat, SUSE, SUSE
- Linux Mint, Scientific Linux
- Raspberry Pi, Raspbian
- Android

Was ist Linux?

Linux ist ein Betriebssystem, das ursprünglich von Linus Torvalds als Neugierprojekt begonnen wurde, aber dann ein Eigenleben annahm – inzwischen arbeiten Hunderte von Entwicklern (nicht nur Studenten und Hobby-Programmierer, sondern auch Profis von Firmen wie IBM, Red Hat oder Oracle) an seiner Weiterentwicklung.

Linux wurde inspiriert von Unix, einem in den 1970er Jahren bei den AT&T Bell Laboratories entwickelten Betriebssystem für »kleine« Computer, das sich schnell zum bevorzugten System für Wissenschaft und Technik entwickelte. Linux verwendet in sehr weiten Teilen dieselben Konzepte und Grundideen wie Unix, und für Unix geschriebene Software ist leicht auf Linux zum Laufen zu bringen, aber Linux selbst enthält keinen Unix-Code, ist also ein unabhängiges Projekt.

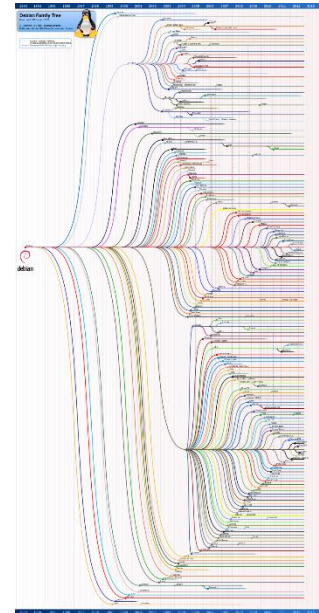
Im Gegensatz zu Windows und OS X steht hinter Linux keine einzelne Firma, deren wirtschaftlicher Erfolg vom Erfolg von Linux abhängt. Linux ist »frei verfügbar« und kann von jedem benutzt werden – auch kommerziell –, der die Spielregeln einhält. Dies zusammen mit dem Umstand, dass Linux inzwischen nicht nur auf PCs läuft, sondern in im Wesentlichen identischer Form vom Telefon (das populärste Betriebssystem für Smartphones, Android, ist ein Ableger von Linux) bis zum größten Großrechner (die 10 schnellsten Rechner der Welt laufen alle unter Linux) auf allen Arten von Rechnern zu finden ist, macht Linux zum flexibelsten Betriebssystem in der Geschichte des modernen Computers.

Distributionen

Eine Linux-Distribution ist eine Zusammenstellung von Software auf Basis des Linux-Kernels.

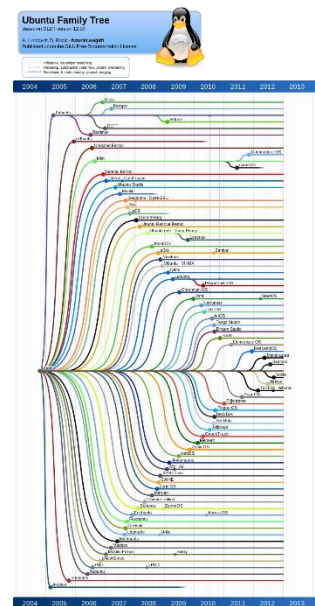
Debian

Debian ist eine Distribution mit einer großen Softwareauswahl und unterstützt mit insgesamt zwölf Prozessor-Architekturen mehr als jede andere binäre Linux-Distribution.



Ubuntu

Ubuntu ist ein Debian-Derivat, das für Heimrechner optimiert wurde.[8] In Ubuntu können proprietäre Treiber nachinstalliert werden.[9] Die Ubuntu-Entwickler veröffentlichen zahlreiche Neuzusammenstellungen mit teilweise fantasievollen Namen. Schlussendlich beruhen alle diese Neuzusammenstellungen nur auf einer unterschiedlichen Auswahl der bei der Erstinstallation enthaltenen Pakete. Basissystem, Installationsprogramm und Repositories sind hingegen identisch. Somit ist es auch möglich, eine Distribution durch Nachinstallieren um den Funktionsumfang einer anderen Neuzusammenstellung zu erweitern.



Weiter Infos zu Linux Distributionen unter folgendem Wikipedia-Artikel:

https://de.wikipedia.org/wiki/Liste_von_Linux-Distributionen

Embedded Systems

Ein Embedded System hat genau definierte Aufgaben; es bildet soft- und hardwaremäßig eine funktionale Einheit, die nur diese definierten Aufgaben erfüllt. Den Anforderungen entsprechend sollten ES-Systeme in vielen Anwendungen einen minimalen Hard- und Software-Aufwand benötigen, sie sollten mit reduzierten Betriebssystemen auskommen, möglichst wenig Komponenten benutzen und einen Prozessor, der die gestellten Aufgaben erfüllt, aber nicht für andere Aufgaben ausgelegt ist. Außerdem sollten sie zur Erhöhung der Performance und Sicherheit Field Programmable Gate Arrays (FPGA) und spezielle ASICs (Application Specific Integrated Circuit) verwenden. Dies alles unter dem Kosten- und Zuverlässigkeitsaspekt aber auch um die Leistungsaufnahme so gering als möglich zu halten. Darüber hinaus spielt die Echtzeitfähigkeit eine entscheidende Rolle.

Embedded Systeme benutzen häufig kein Betriebssystem oder sie arbeiten mit speziellen Embedded-Versionen von Standard-Betriebssystemen, so mit Embedded Windows CE oder Windows XP, Embedded Linux aber auch mit speziellen Betriebssystemen wie OSEK, OS-9 und anderen.

Die Anwendungssoftware, ausgeführt als Embedded Software, zeichnet sich durch feste Funktionen aus und unterstützt nur die entsprechende Anwendung.

Quelle: <https://www.itwissen.info/Embedded-System-embedded-system-ES.html>

Linux in der Clud

Die Flexibilität von Linux macht es auch zum Betriebssystem der Wahl für Anwendungen wie Virtualisierung und »Cloud Computing«. Virtualisierung erlaubt es, auf einem tatsächlichen (»physikalischen«) Rechner mehrere bis viele »virtuelle« Rechner zu simulieren, die über ihr eigenes Betriebssystem verfügen und für dort laufende Programme so aussehen wie »echte« Rechner. Dies führt zu effizienterer Nutzung von Ressourcen und zu höherer Flexibilität: Die gängigen Infrastrukturen für Virtualisierung gestatten es, virtuelle Maschinen sehr schnell von einem physikalischen Rechner auf einen anderen zu »migrieren«, und als Betreiber einer entsprechenden Infrastruktur können Sie so sehr bequem auf Lastsituationen und Ausfälle reagieren. Cloud Computing ist darauf aufbauend die Idee, Rechenleistung nach Bedarf »auf Abruf« zur Verfügung zu stellen und Firmen damit die Möglichkeit zu geben, auf großangelegte Rechenzentren zu verzichten, die nur gelegentlich bei Anfragespitzen tatsächlich voll ausgelastet werden und ansonsten vor allem Kosten verursachen. Anbieter von Cloud-Computing gestatten ihren Kunden die Nutzung virtueller Maschinen über das Internet, bei Abrechnung auf Basis der tatsächlichen Nutzungsdauer, und das kann gegenüber dem Unterhalt eines »realen« Rechenzentrums zu erheblichen Einsparungen führen, insbesondere wenn man einbezieht, dass Sie als Kunde nicht nur die Anschaffungskosten, sondern auch die Personal-, Material- und Stromkosten für den 24/7-Betrieb des Rechenzentrums vermeiden.

Quelle: <https://www.tuxcademy.org/download/de/lxes/lxes-de-manual.pdf> (Seite 28)

3.1.2 Major Open Source Applications

Desktop applications

Text, Tabellen, Präsentation und Datenbank Software

- LibreOffice
- Apache OpenOffice

Beides sind gute alternative zu Standard Office Paket. LibreOffice ist eine Abspaltung von OpenOffice. LibreOffice ist mittlerweile die modernere Version.

Mail Software

Mozilla Thunderbird ist eine Opensource Mail Client. Thunderbird hat mehrere Addons für Kalender, Kontakte oder für das Verschlüsseln von Mails.

Lizenzen: MPL/GPL/LGPL

Webbrowser

Mozilla Firefox ist eine Opensource Webbrowser

Lizenzen: MPL/GPL/LGPL

Bilderbearbeitung Software

Gimp Opensource Bilderbearbeitung Software

Lizenz: LGPLv3+ und GPLv3+

Server applications

Nextcloud & ownCloud

Apache HTTP Server

Apache ist ein der Meistgenutzte Webserver. Der Apache HTTP Server ist in fast allen Linux-Distributionen und in macOS standardmäßig enthalten. Eine beliebte Entwicklungs-Distribution für Windows, Linux und Mac OS X ist XAMPP.

Lizenz: Apache License v2.0

NGINX

Nginx ist ein Opensource Webserver welcher auch als reverse proxy, load balancer, mail proxy und HTTP cache verwendet werden kann.

MariaDB & MySQL

MariaDB

MariaDB ist ein Datenbankmanagementsystem das durch die Abspaltung aus MySQL entstanden ist.

MySQL

MySQL wie ist wie MariaDB Datenbankmanagementsystem.

Lizenz: GPL-2.0-only[\[5\]](#), Proprietäre Lizenz[\[6\]](#)

Network File System (NFS)

Das Network File System (NFS) ist eine Client/Server-Anwendung, die es einem Benutzer ermöglicht, eine Datei auf einem Remote-Computer so anzuzeigen, zu speichern und zu aktualisieren, als ob sie sich auf seinem eigenen Rechner befinden würde

Samba

Samba ist eine beliebte Open-Source-Software, um Server und Desktops mit Linux oder Unix in Umgebungen mit Microsofts Verzeichnisdienst Active Directory zu integrieren. Die Software lässt sich dabei als Controller für die Domäne oder als normales Mitglied verwenden.

Development languages

C

Die Anwendungsbereiche von C sind sehr verschieden. Sie wird zur System- und Anwendungsprogrammierung eingesetzt. Die grundlegenden Programme aller Unix-Systeme und die Systemkernel vieler Betriebssysteme sind in C programmiert. Zahlreiche Sprachen, wie C++, Objective-C, C#, D, Java, JavaScript, PHP, Vala oder Perl, orientieren sich an der Syntax und anderen Eigenschaften von C.

Java

Die Programmiersprache *Java* dient innerhalb der Java-Technologie vor allem zum Formulieren von Programmen. Diese liegen zunächst als reiner, menschenverständlicher [Text](#) vor, dem sogenannten [Quellcode](#). Dieser Quellcode ist nicht direkt ausführbar; erst der Java-Compiler, der Teil des Entwicklungswerkzeugs ist, übersetzt ihn in den maschinenverständlichen Java-Bytecode.

JavaScript

Programmierersprache Java, wurden mithilfe von LiveScript in den Netscape Navigator integriert. Um die Popularität von Java zu nutzen, wurde LiveScript in JavaScript umbenannt, obwohl sich die beiden Sprachen stark voneinander unterscheiden.

Perl

Ursprünglich als Werkzeug zur Verarbeitung und Manipulation von Textdateien insbesondere bei System- und Netzwerkadministration vorgesehen (zum Beispiel Auswertung von Logdateien), hat Perl auch bei der Entwicklung von Webanwendungen und in der Bioinformatik weite Verbreitung gefunden

shell

Eine zeilenweise arbeitende Shell, auch Textshell genannt, stellt dem Benutzer eine Arbeitsumgebung zur Textein- und -ausgabe mit einer sogenannten Befehlszeile (englisch command line) bereit

PHP

PHP ist eine Skriptsprache die hauptsächlich zur Erstellung dynamischer Webseiten oder Webanwendungen verwendet wird.

Package management tools and repositories

Debian Package Manager

dpkg ist die Basis der Paketverwaltung des Betriebssystems Debian und das grundlegende Programm zum Installieren und Manipulieren von Debian-Binärpaketen.

Advanced Packaging Tool

En Paketverwaltungssystem, das im Bereich des Betriebssystems Debian entstanden ist und dpkg zur eigentlichen Paketverwaltung benutzt.

RPM Package Manager

RPM Package Manager ist ein freies (GPL) Paketverwaltungs-System, ursprünglich entwickelt von dem Unternehmen Red Hat.

Yellowdog Updater, Modified

YUM (Yellowdog Updater, Modified) ist ein Paketmanagement-System, das für die Linux-Distribution Yellow Dog Linux entwickelt wurde

3.1.3 Open Source Software and Licensing

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

Wichtige Wissensgebiete:

- Open-Source-Philosophie
- Open-Source-Lizenzierung
- Freie Software Stiftung (FSF), Open Source Initiative (OSI)

Im Folgenden finden Sie eine unvollständige Liste der verwendeten Dateien, Begriffe und Dienstprogramme:

- Copyleft, Permissive
- GPL, BSD, Creative Commons, Creative Commons
- Kostenlose Software, Open Source Software, FOSS, FLOSS
- Open-Source-Geschäftsmodelle

Unterlagen: Linux Essentials von tuxcademy - Kapitel 2

Open source Philosophie

Open source bedeutet quelloffene Software (Quelloffenheit), d.h. eine Software, die unter einer von der Open Source Initiative (OSI) anerkannten Lizenz steht.

Open-Source-Lizenzierung

Sie muss den folgenden charakteristischen Merkmalen entsprechen:

- Die Software (d. h. der Quelltext) liegt in einer für den Menschen lesbaren und verständlichen Form vor.
- Die Software darf beliebig kopiert, verbreitet und genutzt werden.
- Die Software darf verändert und in der veränderten Form weitergegeben werden.

Quelle: <https://www.linux-kurs.eu/opensource.php>

Free Software Foundation

Die Free Software Foundation (FSF, deutsch Stiftung für freie Software) ist eine nichtstaatliche Stiftung, die als gemeinnützige Organisation 1985 von Richard Stallman mit dem Zweck gegründet wurde, freie Software zu fördern und für diese Arbeit Kapital zusammenzutragen. Executive Director der Free Software Foundation ist seit 2011 John Sullivan (Stand: 30. Dezember 2016).

Bis Mitte der 1990er wurden die Finanzmittel der FSF im Wesentlichen dazu verwendet, Programmierer für die Entwicklung freier Software anzustellen. Seitdem viele Unternehmen und Privatpersonen begonnen haben, selbständig freie Software zu schreiben, konzentriert sich die Arbeit der FSF zunehmend auf rechtliche und strukturelle Belange der Freie-Software-Gemeinschaft.

Quelle: https://de.wikipedia.org/wiki/Free_Software_Foundation

Open Source Initiative

Die Open Source Initiative (OSI) ist eine Organisation, die sich der Förderung von Open-Source-Software widmet. Sie wurde im Februar 1998 von Bruce Perens und Eric S. Raymond gegründet. Sie zertifiziert Softwarelizenzen anhand ihrer eigenen Open-Source-Definition. Software, die unter diesen Lizenzen verbreitet wird, darf das Certification Mark der Organisation tragen.

Quelle: https://de.wikipedia.org/wiki/Open_Source_Initiative

Copyleft

Das Copyleft ist eine Klausel in urheberrechtlichen Nutzungslizenzen, die festschreibt, dass Bearbeitungen des Werks nur dann erlaubt sind, wenn alle Änderungen mit mindestens den gleichen oder zumindest ähnlichen Freiheiten weitergegeben werden.

Quelle: <https://de.wikipedia.org/wiki/Copyleft>

Permissive Lizenz

Eine permissive Softwarelizenz, manchmal auch BSD-ähnliche oder BSD-ähnliche Lizenz genannt,[2] ist eine Freie-Software-Lizenz mit minimalen Anforderungen an die Weiterverteilung der Software.

Quelle: https://en.wikipedia.org/wiki/Permissive_software_license

GPL, BSD, Creative Commons

Quellen : <https://www.gnu.org/licenses/license-list.html>

		EULA	GPL	CDDL	BSD
rights in 'copyright'	Right to copy	✗	✓	✓	✓
	Right to modify	✗	*✓	*✓	✓
	Right to distribute	✗	✓	✓	✓
	Right to perform	✓	~	✓	~
	Right to display	✓	~	✓	~
patent rights	Right to manufacture	✗	✓	✓	~
	Right to use	✓	✓	✓	✓
	Right to sell	✗	✓	✓	~

* must use GPL for everything
* use CDDL for things already under CDDL

	Creative Commons License	GPL
	★★★★☆ (45 ratings)	★★★★☆ (55 ratings)
Release Date of the original version	16 December, 2002	January 1989
Released by	Creative Commons, a US non-profit corporation	Free Software Foundation's GNU Project
Guiding Philosophy	Trying to draw a balance between the two extremes of copyrighted work and work in the public domain	To grant users the right to copy, modify, and redistribute the software which would otherwise be prohibited by copyright law
Conditions for licensing	Gives the authors of the creative work, a selection of four conditions and their combinations, under which they license their work	Conditions of licensing are standard and cannot be changed
Type of license	Some of the licenses are Permissive free software licenses	Copyleft license - require copies and derivatives of the source code to be made available on terms no more restrictive than those of the original licence
Use	Creative Commons licenses are for all kinds of creative works: websites, scholarship, music, film, photography, literature, courseware, etc.	Primarily designed for software. One popular example of a GPL-licensed work is WordPress .
What is it?	Creative Commons Licenses are a set of copyright licenses that give the recipients, rights to copy, modify and redistribute the creative material, but giving the authors, the liberty to decide the conditions of licensing	It is the most widely used free software license which grants the recipients, rights to copy, modify and redistribute the software and to ensure that the same rights were preserved in all derivative works
Compatible with GPL	No	Yes
Debian Free Software Guidelines (DFSG) Approval	No	Yes
Open Source Initiative (OSI) Approval	No	Yes
Free Software Foundation (FSF) Approval	Some licenses - Yes, others - No	Yes

Free Software, Open Source Software, FOSS, FLOSS

Die Begriffe Free/Libre Open Source Software bzw. Free and Open Source Software und ihre Akronyme FLOSS bzw. FOSS sind hybride Begriffe für Freie Software und Open-Source-Software. Die Begriffe werden häufig genutzt, um den nicht gelösten Namensstreit zwischen den beiden Bewegungen zu umgehen.

Quelle: https://de.wikipedia.org/wiki/Free/Libre_Open_Source_Software

Ist Open Source Software auch kostenlos?

Nein. Open Source ist das, was wir Ihnen gerade gesagt haben, und es ist nicht notwendig, dass eine Open Source Software kostenlos ist. Sie könnten Ihre Software verkaufen, aber da Sie auch den Quellcode freigeben müssen, berechnen Entwickler meist Geld von den Benutzern für die Softwaredienste und den Support und nicht für die Software, was ein besserer Weg zu sein scheint.

Quelle: <https://fossbytes.com/what-is-free-and-open-source-software-foss-and-why-you-should-use-it/>

Open source business models

Support und Dienstleistungen

Red Hat ist der offensichtlichste Beweis dafür, dass der Verkauf von Open-Source-Software profitabel sein kann. Wie Canonical bietet es seine Software kostenlos an und berechnet Unternehmensnutzer für technische Unterstützungsleistungen. Red Hat verkauft auch Abonnements für seine "Premium"-Distribution Red Hat Enterprise Linux, die Unternehmen durch ihre strengen Tests und ihre Stabilität anspricht. Schließlich verkauft das Unternehmen Unternehmen Unternehmenssoftware-Zertifizierungen, die es Arbeitgebern ermöglichen, hochqualifizierte IT-Fachkräfte zu finden, die von Red Hat zertifiziert wurden und ihre Kenntnisse der Red Hat-Softwaretools nachgewiesen haben.

Werbepartnerschaften

Die Mozilla Corporation ist eine gewinnorientierte Tochtergesellschaft der gemeinnützigen Mozilla Foundation, die Gewinne generiert, die wieder in die Open-Source-Projekte von Mozilla investiert werden. Die Mozilla Corporation erwirtschaftet ihre Einnahmen aus Partnerschaften mit Unternehmen wie Yahoo, Google und Amazon, die sich dafür entschließen, als integrierte Suchoptionen im Firefox-Browser integriert zu werden. Zum Beispiel schloss Yahoo 2014 einen Vertrag mit der Mozilla Corporation, um Yahoo zur Standardsuchmaschine in Firefox zu machen - im Austausch für eine jährliche Zahlung von 375 Millionen Dollar.

Quelle: <https://opensource.com/article/17/12/open-source-business-models>

3.1.4 ICT Skills and Working in Linux

Beschreibung: Grundlegende Kenntnisse der Informations- und Kommunikationstechnologie (IKT) und Arbeiten unter Linux.

Wichtige Wissensgebiete:

- Desktop-Kenntnisse
- So gelangen Sie zur Befehlszeile
- Industrielle Anwendungen von Linux, Cloud Computing und Virtualisierung

Im Folgenden finden Sie eine unvollständige Liste der verwendeten Dateien, Begriffe und Dienstprogramme:

- Verwendung eines Browsers, Datenschutzbedenken, Konfigurationsoptionen, Suche im Web und Speichern von Inhalten
- Terminal und Konsole
- Passwortprobleme
- Datenschutzprobleme und Tools
- Einsatz von gängigen Open-Source-Anwendungen in Präsentationen und Projekten

In Ubuntu kann der Browser einfach über die Favoriten aufgerufen werden (Bild rechts), das Terminal (Die Befehlszeile kann ebenfalls damit aufgerufen werden).



Linux Terminal Cheatsheet:

3.2 Topic 2: Finding Your Way on a Linux System

3.2.1 Command Line Basics

Beschreibung: Grundlagen der Verwendung der Linux-Befehlszeile.

Key Knowledge Areas:

- Basic shell
- Command line syntax
- Variables
- Quoting

Im Folgenden finden Sie eine unvollständige Liste der verwendeten Dateien, Begriffe und Dienstprogramme:

- Bash
- echo
- history
- PATH environment variable
- export
- type

Unterlagen: [Linux Essentials](#) von [tuxcademy](#) - Kapitel 4

3.2.1.1 Basic Shell and Command Line Syntax

Basic Shell Commands:

File Commands	System Info
ls - directory listing ls -al - formatted listing with hidden files cd dir - change directory to <i>dir</i> cd - change to home pwd - show current directory mkdir dir - create a directory <i>dir</i> rm file - delete <i>file</i> rm -r dir - delete directory <i>dir</i> rm -f file - force remove <i>file</i> rm -rf dir - force remove directory <i>dir</i> * cp file1 file2 - copy <i>file1</i> to <i>file2</i> cp -r dir1 dir2 - copy <i>dir1</i> to <i>dir2</i> ; create <i>dir2</i> if it doesn't exist mv file1 file2 - rename or move <i>file1</i> to <i>file2</i> if <i>file2</i> is an existing directory, moves <i>file1</i> into directory <i>file2</i> ln -s file link - create symbolic link <i>link</i> to <i>file</i> touch file - create or update <i>file</i> cat > file - places standard input into <i>file</i> more file - output the contents of <i>file</i> head file - output the first 10 lines of <i>file</i> tail file - output the last 10 lines of <i>file</i> tail -f file - output the contents of <i>file</i> as it grows, starting with the last 10 lines	date - show the current date and time cal - show this month's calendar uptime - show current uptime w - display who is online whoami - who you are logged in as finger user - display information about <i>user</i> uname -a - show kernel information cat /proc/cpuinfo - cpu information cat /proc/meminfo - memory information man command - show the manual for <i>command</i> df - show disk usage du - show directory space usage free - show memory and swap usage whereis app - show possible locations of <i>app</i> which app - show which <i>app</i> will be run by default
Process Management	Compression
ps - display your currently active processes top - display all running processes kill pid - kill process id <i>pid</i> killall proc - kill all processes named <i>proc</i> * bg - lists stopped or background jobs; resume a stopped job in the background fg - brings the most recent job to foreground fg n - brings job <i>n</i> to the foreground	tar cf file.tar files - create a tar named <i>file.tar</i> containing <i>files</i> tar xf file.tar - extract the files from <i>file.tar</i> tar czf file.tar.gz files - create a tar with Gzip compression tar xzf file.tar.gz - extract a tar using Gzip tar cjf file.tar.bz2 - create a tar with Bzip2 compression tar xjf file.tar.bz2 - extract a tar using Bzip2 gzip file - compresses <i>file</i> and renames it to <i>file.gz</i> gzip -d file.gz - decompresses <i>file.gz</i> back to <i>file</i>
File Permissions	Network
chmod octal file - change the permissions of <i>file</i> to <i>octal</i> , which can be found separately for user, group, and world by adding: <ul style="list-style-type: none"> 4 - read (r) 2 - write (w) 1 - execute (x) Examples: chmod 777 - read, write, execute for all chmod 755 - rwx for owner, rx for group and world For more options, see man chmod .	ping host - ping <i>host</i> and output results whois domain - get whois information for <i>domain</i> dig domain - get DNS information for <i>domain</i> dig -x host - reverse lookup <i>host</i> wget file - download <i>file</i> wget -c file - continue a stopped download
SSH	Installation
ssh user@host - connect to <i>host</i> as <i>user</i> ssh -p port user@host - connect to <i>host</i> on port <i>port</i> as <i>user</i> ssh-copy-id user@host - add your key to <i>host</i> for <i>user</i> to enable a keyed or passwordless login	Install from source: ./configure make make install dpkg -i pkg.deb - install a package (Debian) rpm -Uvh pkg.rpm - install a package (RPM)
Searching	Shortcuts
grep pattern files - search for <i>pattern</i> in <i>files</i> grep -r pattern dir - search recursively for <i>pattern</i> in <i>dir</i> command grep pattern - search for <i>pattern</i> in the output of <i>command</i> locate file - find all instances of <i>file</i>	Ctrl+C - halts the current command Ctrl+Z - stops the current command, resume with fg in the foreground or bg in the background Ctrl+D - log out of current session, similar to exit Ctrl+W - erases one word in the current line Ctrl+U - erases the whole line Ctrl+R - type to bring up a recent command !! - repeats the last command exit - log out of current session
* use with extreme caution.	



Quelle: <https://files.fooswire.com/2007/08/fwunixref.pdf>

3.2.1.2 Variablen

Eine Variable ist eine Zeichenkette, der wir einen Wert zuweisen. Der zugewiesene Wert kann eine Zahl, ein Text, ein Dateiname, ein Gerät oder eine andere Art von Daten sein.

Eine Variable ist nichts anderes als ein Zeiger auf die eigentlichen Daten. Die Shell ermöglicht das Erstellen, Zuweisen und Löschen von Variablen.

Variablen können entweder in der Shell oder in Shell-Skripten erstellt werden. Jede Variable, die in einem Shell-Skript erstellt wird, geht verloren, wenn das Skript die Ausführung einstellt. Eine an der Eingabeaufforderung erstellte Variable bleibt jedoch bestehen, bis die Shell beendet ist. Die Syntax für die Erstellung einer Variablen lautet :

```
<variable name> = <value>
```

Weiterführende Informationen:

<https://www.tutorialspoint.com/unix/unix-using-variables.htm>

<https://bash.cyberciti.biz/guide/Variables>

<https://www.w3resource.com/linux-system-administration/shell-variables.php>

3.2.1.3 Quoting

Deine Bash-Shell versteht Sonderzeichen mit besonderen Bedeutungen. Beispielsweise wird \$var verwendet, um den Variablenwert zu erweitern.

Manchmal möchten Sie jedoch keine Variablen oder Wildcards verwenden. Sie sollten z.B. nicht den Wert von \$PATH ausgeben, sondern einfach \$PATH auf dem Bildschirm als Wort ausgeben. Sie können die Bedeutung eines Sonderzeichens aktivieren oder deaktivieren, indem Sie es in einfache Anführungszeichen setzen. Dies ist auch nützlich, um Warnungen und Fehlermeldungen beim Schreiben der Shell-Skripte zu unterdrücken.

Quote type	Name	Meaning	Example (type at shell prompt)
"	The double quote	The double quote ("quote") protects everything enclosed between two double quote marks except \$, ', " and \. Use the double quotes when you want only variables and command substitution . * Variable - Yes * Wildcards - No * Command substitution - yes	The double quotes allows to print the value of \$SHELL variable, disables the meaning of wildcards , and finally allows command substitution. <pre>echo "\$SHELL" echo "/etc/*.conf" echo "Today is \$(date)"</pre>
'	The single quote	The single quote ('quote') protects everything enclosed between two single quote marks. It is used to turn off the special meaning of all characters. * Variable - No * Wildcards - No * Command substitution - No	The single quotes prevents displaying variable \$SHELL value, disabled the meaning of wildcards /etc/*.conf, and finally command substitution (\$date) itself. <pre>echo '\$SHELL' echo '/etc/*.conf' echo 'Today is \$(date)'</pre>
\	The Backslash	Use backslash to change the special meaning of the characters or to escape special characters within the text such as quotation marks.	You can use \ before dollar sign to tell the shell to have no special meaning. Disable the meaning of the next character in \$PATH (i.e. do not display value of \$PATH variable): <pre>echo "Path is \\$PATH" echo "Path is \$PATH"</pre>

Weiterführende Informationen:

<https://bash.cyberciti.biz/guide/Quoting>

<https://www.tutorialspoint.com/unix/unix-quoting-mechanisms.htm>

3.2.2 Using the Command Line to Get Help

Weight: 2

Description: Running help commands and navigation of the various help systems.

Key Knowledge Areas:

- Man pages
- Info pages

The following is a partial list of the used files, terms and utilities:

- man
- info
- /usr/share/doc/
- locate

Unterlagen: [Linux Essentials](#) von [tuxcademy](#) - Kapitel 5

3.2.2.1 Man Pages

Linux-Manualseiten gibt es von Anfang an und helfen zu erklären, was jeder Befehl kann. Jede Manpage sollte dir einen allgemeinen Überblick über den Befehl, seine Flags (Optionen) und möglicherweise ein Beispiel für die Verwendung nach unten geben.

```
DF(1)                                User Commands                                DF(1)
NAME
    df - report file system disk space usage
SYNOPSIS
    df [OPTION]... [FILE]...
DESCRIPTION
    This manual page documents the GNU version of df. df displays the
    amount of disk space available on the file system containing each file
    name argument. If no file name is given, the space available on all
    currently mounted file systems is shown. Disk space is shown in 1K
    blocks by default, unless the environment variable POSIXLY_CORRECT is
    set, in which case 512-byte blocks are used.

    If an argument is the absolute file name of a disk device node contain-
    ing a mounted file system, df shows the space available on that file
    system rather than on the file system containing the device node. This
    version of df cannot show the space available on unmounted file sys-
    tems, because on most kinds of systems doing so requires very non-
    portable intimate knowledge of file system structures.
OPTIONS
    Show information about the file system on which each FILE resides, or
    all file systems by default.

    Mandatory arguments to long options are mandatory for short options
    too.

    -a, --all
        include dummy file systems
Manual page df(1) line 1 (press h for help or q to quit)
```

Die Manual-Page wird
folgendermassen aufgerufen:

```
<Programm> --manual
```

Weiter Informationen:

https://en.wikipedia.org/wiki/Man_page

3.2.2.2 Info Pages

Auf Unix-ähnlichen Betriebssystemen liest der Befehl `info` die Dokumentation, die in dem von der Free Software Foundation entwickelten Info-Format gespeichert ist.

Beschreibung

Das Informationsformat ähnelt dem des «man», dem traditionellen Unix-Manual-Format. Info bietet zusätzlich eine robuste Struktur für die Verknüpfung von Seiten untereinander, ähnlich wie bei Hyperlinks. Info-Seiten werden mit den texinfo-Tools erstellt und können mit anderen Seiten verlinkt werden, Menüs erstellen und die Navigation im Allgemeinen erleichtern.

Der Standardort der Infodokumentation ist `/usr/share/info`.

Syntax

```
info [OPTION]... [MENU-ITEM...]
```

Examples

```
info emacs
```

Start at emacs node from top-level dir.

```
info --show-options emacs
```

start at node with emacs command line options

Weiterführende Informationen :

https://www.linuxtopia.org/online_books/introduction_to_linux/linux_The_Info_pages.html

<https://www.computerhope.com/unix/info.htm>

3.2.3 Using Directories and Listing Files

Weight: 2

Description: Navigation of home and system directories and listing files in various locations.

Key Knowledge Areas:

- Files, directories
- Hidden files and directories
- Home directories
- Absolute and relative paths

The following is a partial list of the used files, terms and utilities:

- Common options for ls
- Recursive listings
- cd
- . and ..
- home and ~

Unterlagen:

- [Linux Essentials](#) von [tuxcademy](#) - Kapitel 6 und Kapitel 10
- [Filesystem](#)

3.2.3.1 Files, directories, Hidden Files and Hidden directories

«ls» is a Linux shell command that lists directory contents of files and directories.

ls syntax

```
$ ls [options] [file|dir]
```

ls command options

option	description
ls -a	list all files including hidden file starting with '.'
ls --color	colored list [=always/never/auto]
ls -d	list directories - with '*'
ls -F	add one char of */=>@ to enteries
ls -i	list file's inode index number
ls -l	list with long format - show permissions
ls -la	list long format including hidden files
ls -lh	list long format with readable file size
ls -ls	list with long format with file size
ls -r	list in reverse order
ls -R	list recursively directory tree
ls -s	list file size
ls -S	sort by file size
ls -t	sort by time & date
ls -X	sort by extension name

Weiterführende Informationen :

<https://www.rapidtables.com/code/linux/ls.html>

3.2.3.2 Absolute and relative paths

WHAT IS A PATH?

A path is a unique location to a file or a folder in a file system of an OS. A path to a file is a combination of / and alpha-numeric characters.

WHAT IS AN ABSOLUTE PATH?

An absolute path is defined as the specifying the location of a file or directory from the root directory(/). In other words we can say absolute path is a complete path from start of actual filesystem from / directory.

SOME EXAMPLES OF ABSOLUTE PATH:

```
/var/ftp/pub  
/etc/samba.smb.conf  
/boot/grub/grub.conf
```

WHAT IS THE RELATIVE PATH?

Relative path is defined as path related to the present working directory(pwd). Suppose I am located in /var/log and I want to change directory to /var/log/kernel. I can use relative path concept to change directory to kernel.

changing directory to /var/log/kernel by using relative path concept.

```
pwd  
/var/log  
cd kernel
```

Note: If you observe there is no / before kernel which indicates it's a relative directory to present working directory.

Changing directory to /var/log/kernel using absolute path concept.

```
cd /var/log/kernel
```

Note: We can use an absolute path from any location where as if you want to use relative path we should be present in a directory where we are going to specify relative to that present working directory.

Weiterführende Informationen :

<https://www.linuxnix.com/abslute-path-vs-relative-path-in-linuxunix/>

3.2.4 Creating, Moving and Deleting Files

Weight: 2

Description: Create, move and delete files and directories under the home directory.

Key Knowledge Areas:

- Files and directories
- Case sensitivity
- Simple globbing

The following is a partial list of the used files, terms and utilities:

- mv, cp, rm, touch
- mkdir, rmdir

Unterlagen:

- [Linux Essentials](#) von [tuxcademy](#) - Kapitel 6 (letzte zwei Kapitel)
- [Files](#)

3.2.4.1 mkdir - Create new Folders

mkdir

Unter Linux wird häufig `mkdir (/bin/mkdir)` verwendet, der Befehl zum Erstellen von Verzeichnissen. Ich finde oft, dass die Leute überrascht sind, dass die Option `-m` da ist und nicht wusste, dass man das Verzeichnis erstellen und die Berechtigungen in einem setzen kann.

Beispiele

Ein einzelnes Verzeichnis erstellen:

```
mkdir ubuntu
```

Dieses Beispiel erstellt im aktuellen Verzeichnis das Verzeichnis `ubuntu`.

Das Erstellen mehrerer Verzeichnisse funktioniert so:

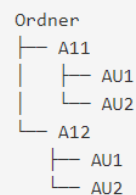
```
mkdir ubuntu linux
```

Dieses Beispiel erstellt im aktuellen Verzeichnis die zwei Verzeichnisse namens `ubuntu` und `linux`.

Folgendermassen ist es möglich Ordner inklusive Unterordnern zu erstellen:

```
mkdir -p Ordner/{A11,A12}/{AU1,AU2}
```

Dies würde zur folgenden Ordnerstruktur führen: →

**Options:**

`-m=mode,`
`--mode=mode`

You can use the `-m` option to set a file mode ([permissions](#), etc.) for the created directories. The syntax of *mode* is the same as with the `chmod` command.

`-p,`
`--parents`

Create [parent](#) directories as necessary. When this option is specified, no error is reported if a *directory* already exists.

`-v,`
`--verbose`

[Verbose](#) output. Print a message for each created directory.

`-Z=context,`
`--context=context`

If you are using [SELinux](#), this option sets the security context of each created directory to *context*. For detailed information about security contexts, consult your SELinux documentation.

`--help`

Display a help message, and exit.

`--version`

Display [version](#) information, and exit.

3.2.4.2 touch – Create files

Eine der größten Anwendungen des Touch-Befehls unter Linux ist das Erstellen einer neuen leeren Datei. Die Syntax ist sehr einfach.

```
Touch filename
```

Natürlich können ebenfalls Dateien mit dem Texteditor erstellt werden. Z.B. NANO:

```
Nano filename.txt
```

3.2.4.3 mv – Move and Rename Files

Der Befehl mv macht eine Sache - er verschiebt eine Datei von einem Ort zum anderen. Dies kann etwas irreführend sein, da mv auch zum Umbenennen von Dateien verwendet wird. Wie? Ganz einfach. Hier ist ein Beispiel. Angenommen, Sie haben die Datei testfile in /home/jack/ und möchten sie in testfile2 umbenennen (wobei Sie sie am selben Ort behalten). Um dies zu tun, würdest du den Befehl mv wie folgt verwenden:

```
mv /home/jack/testfile /home/jack/testfile2
```

oder, wenn du bereits bei /home/jack bist:

```
mv testfile testfile2
```

Die obigen Befehle würden /home/jack/testfile nach /home/jack/testfile2 verschieben und die Datei effektiv umbenennen. Aber was wäre, wenn Sie die Datei einfach verschieben wollten?

Angenommen, Sie wollen Ihr Home-Verzeichnis (in diesem Fall /home/jack) frei von verstreuten Dateien halten. Du kannst diese Testdatei mit dem Befehl in /home/jack/Documents verschieben:

```
mv /home/jack/testfile /home/jack/Documents/
```


3.2.4.4 Delete Files and Directories

Deleting a file

To delete a single file using the *rm* command, use the syntax below :

```
$ rm filename
```

Deleting multiple files

To remove many files in a single command using the *rm* command, use the syntax :

```
$ rm file1 file2 file2
```

Force remove files

To remove files forcefully, especially ones that have been write-protected, use the *-f* flag as shown below :

```
$ rm -f file1.txt
```

Removing empty directories

If you want to remove an empty directory, use the *-d* flag with the *rm* command

```
$ rm -d directory_name
```

Deleting non-empty directories

If you want to remove non-empty directories together with all files, use the *-r* flag as shown

```
$ rm -r directory_name
```

3.3.1 Topic 3: The Power of the Command Line

Archiving Files on the Command Line

Weight: 2

Description: Archiving files in the user home directory.

Key Knowledge Areas:

- Files, directories
- Archives, compression

The following is a partial list of the used files, terms and utilities:

- tar
- Common tar options
- gzip, bzip2, xz
- zip, unzip

Unterlagen:

- [Linux Essentials](#) von [tuxcademy](#) - Kapitel 11

3.3.1.1 Archiving Files and Directories

In Linux-Systemen können Sie mit dem Befehl `tar` Dateien auf einem Gerät, wie beispielsweise einer Festplatte oder einem Band, archivieren. Das Tar-Programm unter Linux erstellt eine Archivdatei, die andere Verzeichnisse und Dateien enthalten kann und komprimiert das Archiv (optional) für eine effiziente Speicherung. Anschließend wird das Archiv auf ein bestimmtes Gerät oder eine andere Datei geschrieben. Viele Softwarepakete werden in Form einer komprimierten Tar-Datei verteilt.

Die Befehlssyntax des Tar-Programms unter Linux lautet wie folgt:

```
tar options destination source
```

Hier werden Optionen normalerweise durch eine Folge von Einzelbuchstaben spezifiziert, wobei jeder Buchstabe angibt, was `tar` tut; `destination` ist der Gerätenamen des Backups; und `Quelle` ist eine Liste von Datei- oder Verzeichnisnamen, die die zu sichernden Dateien bezeichnen.

Sichern und Wiederherstellen eines Einzelvolumenarchivs unter Linux

Angenommen, Sie möchten den Inhalt des Verzeichnisses `/etc/X11` auf einer Festplatte sichern. Melden Sie sich als `root` an und geben Sie den folgenden Befehl ein, wobei `xxx` Ihr Laufwerk darstellt:

```
tar zcvf /dev/xxx /etc/X11
```

Das Tar-Programm zeigt eine Liste von Dateinamen an, wenn jede Datei in das komprimierte Tar-Archiv kopiert wird. In diesem Fall sind die Optionen `zcvf`, das Ziel ist `/dev/xxx` (das Laufwerk) und die Quelle ist das Verzeichnis `/etc/X11` (was alle seine Unterverzeichnisse und deren Inhalt beinhaltet). Sie können einen ähnlichen `tar`-Befehl verwenden, um Dateien auf ein Band zu sichern, indem Sie den Speicherort der Festplatte durch den der Bandstation ersetzen, z.B. `/dev/st0` für ein SCSI-Bandlaufwerk.

Option	Does the Following
<code>c</code>	Creates a new archive.
<code>f</code>	Specifies the name of the archive file or device on the next field in the command line.
<code>M</code>	Specifies a multivolume archive.
<code>t</code>	Lists the contents of the archive.
<code>v</code>	Displays verbose messages.
<code>x</code>	Extracts files from the archive.
<code>z</code>	Compresses the tar archive by using <code>gzip</code> .

Um den Inhalt des tar-Archivs anzuzeigen, das Sie auf dem Laufwerk erstellen, geben Sie den folgenden Befehl ein (xxx durch das Laufwerk ersetzen):

```
tar ztf /dev/xxx
```

Sie sehen eine Liste von Dateinamen (die jeweils mit /etc/X11 beginnen), die anzeigen, was sich in der Sicherung befindet. In diesem tar-Befehl listet die Option t den Inhalt des tar-Archivs auf.

Um die Dateien aus einem tar-Backup zu extrahieren, führen Sie diese Schritte aus, während Sie als root angemeldet sind:

1. Ändern Sie das Verzeichnis to/tmp, indem Sie diesen Befehl eingeben:

```
cd /tmp
```

In diesem Schritt können Sie das Extrahieren der Dateien aus dem Tar-Backup üben. Für ein echtes Backup ändern Sie das Verzeichnis an einen geeigneten Ort. (Typischerweise geben Sie cd /.)

2. Geben Sie den folgenden Befehl ein:

```
tar zxvf /dev/xxx
```

Dieser tar-Befehl verwendet die Option x, um die Dateien aus dem auf dem Gerät gespeicherten Archiv zu extrahieren (xxx durch das Laufwerk ersetzen).

Wenn Sie nun den Inhalt des Verzeichnisses /tmp überprüfen, stellen Sie fest, dass der Befehl tar einen Verzeichnisbaum etc/X11 in /tmp erstellt und alle Dateien aus dem tar-Archiv in dieses Verzeichnis zurückstellt. Der Befehl tar entfernt den führenden / aus den Dateinamen im Archiv und stellt die Dateien im aktuellen Verzeichnis wieder her. Wenn Sie das Verzeichnis /etc/X11 aus dem Archiv wiederherstellen möchten, verwenden Sie diesen Befehl (anstelle des Gerätenamens xxx):

```
tar zxvf /dev/xxx -C /
```

Die Option -C ändert Verzeichnisse in das angegebene Verzeichnis (in diesem Fall das Stammverzeichnis von /), bevor Sie das tar machen; das / am Ende des Befehls bezeichnet das Verzeichnis, in dem Sie die Sicherungsdateien wiederherstellen möchten.

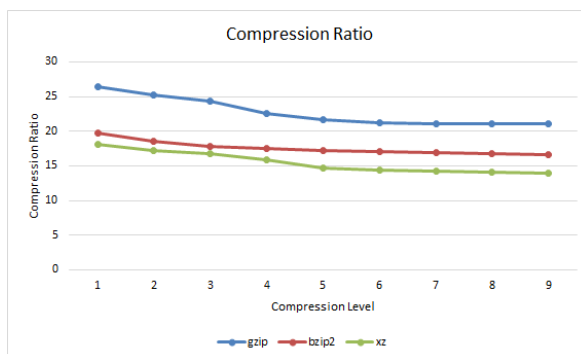
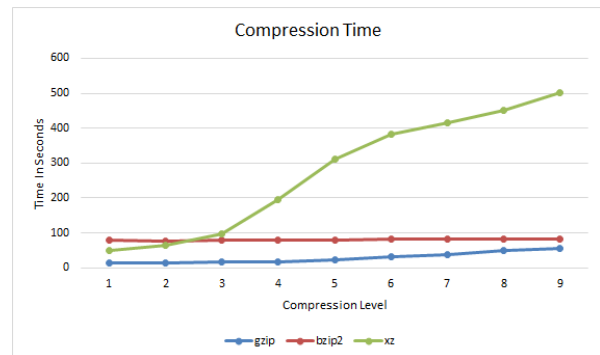
In Linux-Systemen können Sie mit dem Befehl tar ein Archiv erstellen, anzeigen und wiederherstellen. Sie können das Archiv in einer Datei oder auf jedem von Ihnen angegebenen Gerät mit einem Gerätenamen speichern.

Weiterführende Informationen:

<https://www.dummies.com/computers/operating-systems/linux/how-to-archive-files-and-directories-using-tar-in-linux/>

3.3.1.2 Different Compression Methods

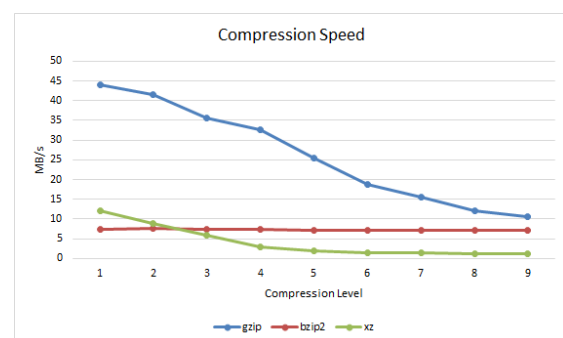
If you are interactively compressing files on the fly then you may want to do this quickly with `gzip -6` (default compression level) or `xz -1`, however if you're configuring log rotation which will run automatically over night during a low resource usage period then it may be acceptable to use more CPU resources with `xz -9` to save the greatest amount of space possible. For instance kernel.org compress the Linux kernel with xz, in this case spending extra time to compress the file well once makes sense when it will be downloaded and decompressed thousands of times resulting in bandwidth savings yet still decent decompression speeds.



Based on the results here, if you're simply after being able to compress and decompress files as fast as possible with little regard to the compression ratio, then gzip is the tool for you. If you want a better compression ratio to save more disk space and are willing to spend extra processing time to get it then xz will be best to use. Although xz takes the longest to compress at higher compression levels, it has a fairly good decompression speed and compresses quite fast

at lower levels. Bzip2 provides a good trade off between compression ratio and processing speed however it takes the longest to decompress so it may be a good option if the content that is being compressed will be infrequently decompressed.

In the end the best option will come down to what you're after between processing time and compression ratio. With disk space continually becoming cheaper and available in larger sizes you may be fine with saving some CPU resources and processing time to store slightly larger files. Regardless of the tool that you use, compression is a great resource for saving storage space.



Quelle : <https://www.rootusers.com/gzip-vs-bzip2-vs-xz-performance-comparison/>

3.3.2 Searching and Extracting Data from Files

Weight: 3

Description: Search and extract data from files in the home directory.

Key Knowledge Areas:

- Command line pipes
- I/O redirection
- Basic Regular Expressions using ., [, *, and ?

The following is a partial list of the used files, terms and utilities:

- grep
- less
- cat, head, tail
- sort
- cut
- wc

Unterlagen:

- [Linux Essentials](#) von [tuxcademy](#) - Kapitel 8
- [Standard-Datenströme](#)
- [File Manipulationen](#)

cat (Kapitel 8.3.1)

cat ist ursprünglich zum Zusammenfügen von Dateien gedacht (von concatenate = verketten, verknüpfen).

```
cat >main.js <<%EOF%
    console.log('Hallo Welt');
%EOF%
ssh <server> tar czf - /etc | cat - >sicherung.tgz
```

head, tail, less (Kapitel 8.3.2)

head + tail - Anfang und Ende von Dateien

```
head /proc/cpuinfo
tail /var/log/apache2/error.log
tail -f /var/log/apache2/access.log
```

less oder more - Zeigt Texte (etwa Handbuchseiten) seitenweise an

```
less /proc/cpuinfo
```

wc, sort, uniq (Kapitel 8.3.1)

wc – zählt die Anzahl Wörter, Zeilen

```
wc /proc/cpuinfo  
ls -l | wc -l
```

sort - Sortiert die Zeilen seiner Eingabe

```
ls | sort
```

uniq - Entfernt doppelte Einträge

```
cat <meineDaten> | sort | uniq
```

cut (Kapitel 8.4.2)

cut - Extrahiert Felder oder Spalten aus seiner Eingabe

```
cut -d: -f1 /etc/passwd | sort  
cut -c1-3 /etc/passwd | sort | uniq | wc -l
```

siehe auch: <https://wiki.ubuntuusers.de/cut/>

Reguläre Ausdrücke (Kapitel 7)

Ein regulärer Ausdruck (englisch regular expression) ist eine Zeichenkette, die der Beschreibung von Mengen von Zeichenketten mit Hilfe bestimmter syntaktischer Regeln dient.

```
^ - Beginn Zeile, $ - Ende Zeile  
[] - Ausdruck beinhaltet Zeichen, z.B. [A-Z]
```

Wildcards: * - beliebige Wiederholungen, . – ein beliebiges Zeichen.

grep (Kapitel 7.3)

grep, fgrep, egrep - Sucht in Dateien nach Zeilen mit bestimmtem Inhalt

```
grep ubuntu /etc/passwd  
grep -v ubuntu /etc/passwd  
grep ^[a-z] /etc/passwd
```

siehe auch: <https://wiki.ubuntuusers.de/grep/>

3.3.3 Turning Commands into a Script

Weight: 4

Description: Turning repetitive commands into simple scripts.

Key Knowledge Areas:

- Basic shell scripting
- Awareness of common text editors (vi and nano)

The following is a partial list of the used files, terms and utilities:

- #! (shebang)
- /bin/bash
- Variables
- Arguments
- for loops
- echo
- Exit status

Unterlagen:

- [Linux Essentials](#) von [tuxcademy](#) - Kapitel 9
- [Prozesse](#)
- [sudo](#)
- [vi Text Editor](#)

3.3.3.1 #! (shebang)

The #! syntax used in scripts to indicate an interpreter for execution under UNIX / Linux operating systems. Most Linux shell and [perl](#) / [python](#) script starts with the following line:

```
#!/bin/bash
```

OR

```
#!/usr/bin/perl
```

OR

```
#!/usr/bin/python
```

3.3.3.2 Variables

Eine Umgebungs-Variable (Environment-Variable) wird z.B. für Komfortfunktionen in der Shell verwendet, um Suchpfade zu Programmen oder eine zentrale Proxy-Konfiguration zu setzen. Environment-Variablen werden bei der Prozessgenerierung "vererbt", d.h. Kindprozesse bekommen eine Kopie des Environments des Vaterprozesses und vererben diese auch wieder an ihre Kinder weiter.

Es gibt unterschiedliche Möglichkeiten, Variablen zu definieren; das ist vom beabsichtigten Zweck abhängig. Definiert man Variablen nur für die aktuelle Sitzung, dann ist sie nur bis zum nächsten Neustart gültig.

```
VARIABLE=Wert-der-Variable
```

3.3.3.3 Argumente

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. Im Gegensatz zu Optionen kann es häufig eine sehr große Zahl verschiedener Argumente geben. Optionen hingegen sind immer nur in relativ beschränkter Zahl verfügbar - immer gerade so viele, wie der Programmierer in sein Programm implementiert hat. Nebenbei bemerkt ist jedoch auch die Anzahl der Argumente einer Kommandozeile nicht unbeschränkt, denn die Argumentzeile eines Kommandos darf eine Größe von 128 Kilobyte nicht überschreiten.

3.3.3.4 for-Loops

Eine 'for-Schleife' ist eine bash-Programmierspracheanweisung, die es ermöglicht, Code wiederholt auszuführen. Eine for-Schleife wird als Iterationsanweisung klassifiziert, d.h. es ist die Wiederholung eines Prozesses innerhalb eines Bash-Skripts. Beispielsweise können Sie den UNIX-Befehl oder die Aufgabe 5 mal ausführen oder die Liste der Dateien mit einer for-Schleife lesen und verarbeiten. Eine for-Schleife kann an einem Shell-Prompt oder in einem Shell-Skript selbst verwendet werden.

Beispiel:

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

3.3.3.5 Echo

«echo» ist ein integrierter Befehl in den Bash- und C-Shells, der seine Argumente in die Standardausgabe schreibt.

Die Syntax für echo lautet:

```
echo [option(s)] [string(s)]
```

Options

These options may be specified before the string, and affect the behavior of **echo**.

-n	Do not output a trailing newline .
-e	Enable interpretation of backslash escape sequences (see below for a list of these).
-E	Disable interpretation of backslash escape sequences. This is the default.

Options

If a long option is specified, you may not specify a string to be echoed. These options are for getting information about the program only.

--help	Display a help message and exit.
--version	Output version information and exit.

Escape sequences

If you specify the **-e** option, the following escape sequences are recognized in your string:

Sequence	Interpreted as
\\	A literal backslash character (" \ ").
\a	An alert (The BELL character).
\b	Backspace.
\c	Produce no further output after this.
\e	The escape character; equivalent to pressing the escape key.
\f	A form feed .
\n	A newline.
\r	A carriage return .
\t	A horizontal tab.
\v	A vertical tab.
\0NNN	byte with octal value <i>NNN</i> (which can be 1 to 3 digits).
\xHH	byte with hexadecimal value <i>HH</i> (which can be either 1 or 2 digits)

3.3.3.6 Exit and Exit Status

Der Exit-Befehl beendet ein Skript, genau wie in einem C-Programm. Es kann auch einen Wert zurückgeben, der dem übergeordneten Prozess des Skripts zur Verfügung steht.

Jeder Befehl gibt einen Ausgangsstatus zurück (manchmal auch als Rückgabestatus oder Ausgangscode bezeichnet). Ein erfolgreicher Befehl gibt eine 0 zurück, während ein erfolgloser Befehl einen Wert ungleich Null zurückgibt, der normalerweise als Fehlercode interpretiert werden kann. Ausgeklügelte UNIX-Befehle, Programme und Dienstprogramme geben nach erfolgreicher Beendigung einen 0-Exit-Code zurück, obwohl es einige Ausnahmen gibt.

3.4 Topic 4: The Linux Operating System

3.4.1 Choosing an Operating System

Weight: 1

Description: Knowledge of major operating systems and Linux distributions.

Key Knowledge Areas:

- Differences between Windows, OS X and Linux
- Distribution life cycle management

The following is a partial list of the used files, terms and utilities:

- GUI versus command line, desktop configuration
- Maintenance cycles, beta and stable

Unterlagen: [Linux Essentials](#) von [tuxcademy](#) - Kapitel 1

3.4.1.1 Differences between Windows, OS X and Linux

Key Differences Between Linux and Windows Operating System

- Linux is free and open source operating system whereas Windows is a commercial operating system whose source code is inaccessible.
- Windows is not customizable as against Linux is customizable and a user can modify the code and can change its the look and feel.
- Linux provides high security than windows because Linux is open source.
- Windows must boot from the primary partition. In contrast, there is no such constraint in Linux it can be booted from either primary or logical partition.
- The separation of the directories is done using a backslash in windows. On the contrary, in Linux, these are separated by using forward slash.
- In Linux, file names are case sensitive while windows file name are case-insensitive.
- Linux uses the monolithic kernel which consumes more running space whereas Windows uses the micro kernel which takes less space but system running efficiency is lower than Linux.

Mac OS and Linux are very similar; both have roots in Unix, a simple but powerful and more secure operating system. Mac OS is proprietary, and it runs on their hardware, which jacks up the price.

3.4.1.2 Distribution life cycle management

Linux distributions tend to use two different types of release cycles: standard releases and rolling releases. Some people swear by rolling releases to have the latest software, while others like standard releases for being more stable and tested.

To understand the difference, you need to know how Linux distributions are put together. They contain software from many different projects — the Linux kernel, GNU shell utilities, Xorg X server, GNOME desktop environment, and LibreOffice office suite are all developed by different software projects with different development cycles. It's the job of a Linux distribution to take all this software in source code form, compile it, package it into easily installable software packages, test it to ensure it works together, and release a complete package of software we call a "Linux distribution."

Linux distributions — whether they use a standard release cycle or a rolling release cycle — all take their software and package it up into software packages that they distribute to users. The difference is in how they distribute new versions of these packages.

A Standard Release Cycle

Most Linux distributions use standard release cycles. For example, Ubuntu uses standard releases — these may also be called point releases or stable releases. The Ubuntu project regularly release new versions of Ubuntu every six months. During the six-month development process, they take the latest versions of all the software in their repositories and package it up, updating all the software. They then "freeze" the versions of the software in the Ubuntu repositories and spend a few months testing it, making sure all the software versions work well together and fixing bugs.

When a new version of Ubuntu is released, the software in it has been tested to ensure it works well together. This release stays frozen in time as much as possible. Ubuntu releases updated software versions to fix security problems and other important bugs, but they won't just update software to add new features or bump the version number.

A Rolling Release Cycle

A rolling release cycle dispenses with regular, standard Linux distribution releases. For example, Arch Linux uses a rolling release cycle. There aren't multiple different releases of Arch. Instead, there's just a single version of Arch. Software packages are tested and then released immediately to the stable version of the Linux distribution. Depending on your distribution, they may not even see much testing before they're released as stable updates. When a new version of an application or system utility is released, it will head straight to the current Linux distribution. A rolling release distribution is never "frozen in time" — instead, it's updated on a rolling basis.

Quelle : <https://www.howtogeek.com/192939/linux-distribution-basics-rolling-releases-vs.-standard-releases/>

3.4.2 Understanding Computer Hardware

Weight: 2

Description: Familiarity with the components that go into building desktop and server computers.

Key Knowledge Areas:

- Hardware

The following is a partial list of the used files, terms and utilities:

- Motherboards, processors, power supplies, optical drives, peripherals
- Hard drives, solid state disks and partitions, /dev/sd*
- Drivers

Unterlagen: [Linux Essentials](#) von [tuxcademy](#) - Kapitel 1

3.4.2.1 Aus was besteht ein Computer?

Ein Computer besteht aus vielen einzelnen Komponenten, welche alle zusammenarbeiten. Im Nachfolgenden erkläre ich die einzelnen Bestandteile. Für alle Bestandteile gibt es Treiber und Firmwares, welche der CPU mitteilen, wie sie zu bedienen sind.

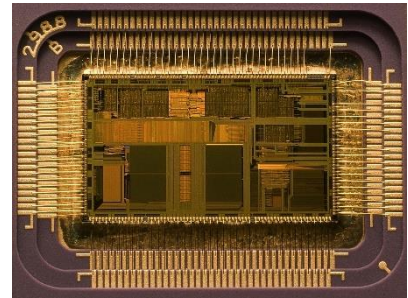
Mainboard

Das Mainboard ist die Hauptplatine des Computers. Auf ihr befinden sich alle Anschlüsse für die benötigten Komponenten eines Systems.



CPU

Die CPU ist die zentrale Steuereinheit des Computers. Sie verarbeitet Befehle und ist für die gesamte Steuerung des Systems verantwortlich.



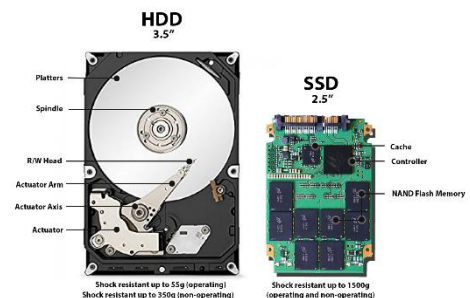
RAM

Der RAM (Arbeitsspeicher) ist das Kurzzeitgedächtnis des PC's. In ihm werden für die Ausführung von Programmen benötigte Dateien gespeichert. Nach dem Herunterfahren wird dieser Speicher jeweils wieder gelöscht.



HDD und SSD

Die Harddisk wie auch die SSD sind Speichermedien. Sie speichern Daten ab solange man möchte. Sie werden auch nach dem herunterfahren nicht einfach gelöscht. Auf ihnen befindet sich das Betriebssystem. Der Unterschied dieser zwei Speichermedien ist die Hardware selbst. Bei einer HDD ist eine oder meist mehrere Magnetscheiben vorhanden, auf welche die Daten geschrieben werden. Bei einer SSD werden Daten auf Flash Memory gespeichert, ähnlich dem eines USB-Sticks.



Grafikkarte

Die Grafikkarte ist hauptsächlich für die Bildverarbeitung zuständig. Sie wandelt Daten in Bilder um, berechnet Spielegrafik und kann auch als Recheneinheit für spezifische wissenschaftliche Operationen eingesetzt werden.



PSU

Das Netzteil wird benötigt, um den Strom für den Betrieb bereitzustellen.



DVD Laufwerk

Das DVD-Laufwerk dient der Wiedergabe von CD's, DVD's usw. Auf den meisten Geräten ist es auch möglich DVD's oder CD's zu brennen.



Peripheriegeräte

Das Peripheriegerät ist eine Komponente oder ein Gerät, das sich außerhalb der Zentraleinheit eines Computers befindet (kurz: Zubehör). Vereinfacht kann zwischen im Computer verbauten (internen) und mit diesem durch ein Kabel (oder auch durch Infrarot- oder Funktechnik) verbundenen (externen) Peripheriegeräten unterschieden werden.



Peripheriegeräte dienen der Ein- und Ausgabe von Daten oder Befehlen in die Zentraleinheit. Sie erbringen eine „Dienstleistung“ für den Benutzer, diese kann entweder sichtbar (z. B. Papier-Ausdruck) oder unsichtbar (z. B. Signalwandlung interner Modems) erfolgen.

3.4.3 Where Data is Stored

Weight: 3

Description: Where various types of information are stored on a Linux system.

Key Knowledge Areas:

- Programs and configuration
- Processes
- Memory addresses
- System messaging
- Logging

The following is a partial list of the used files, terms and utilities:

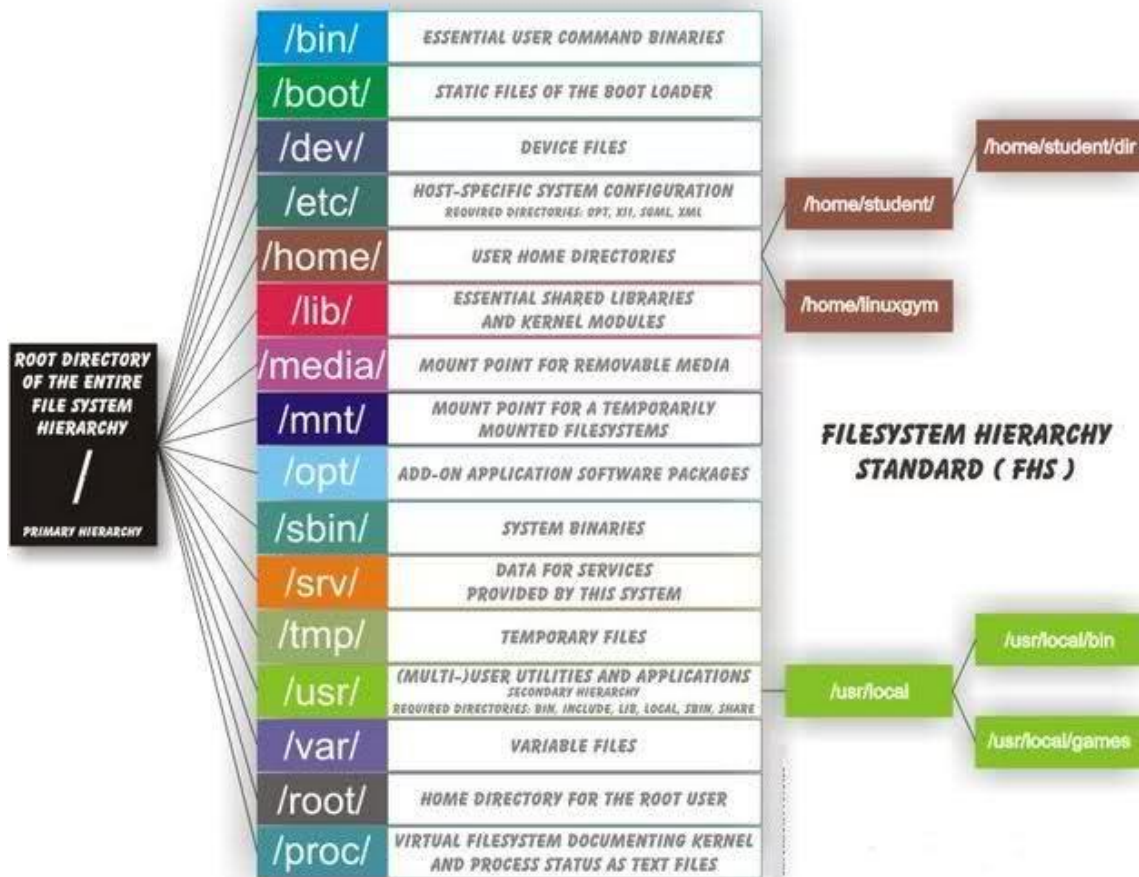
- ps, top, free
- syslog, dmesg
- /etc/, /var/log/
- /boot/, /proc/, /dev/, /sys/

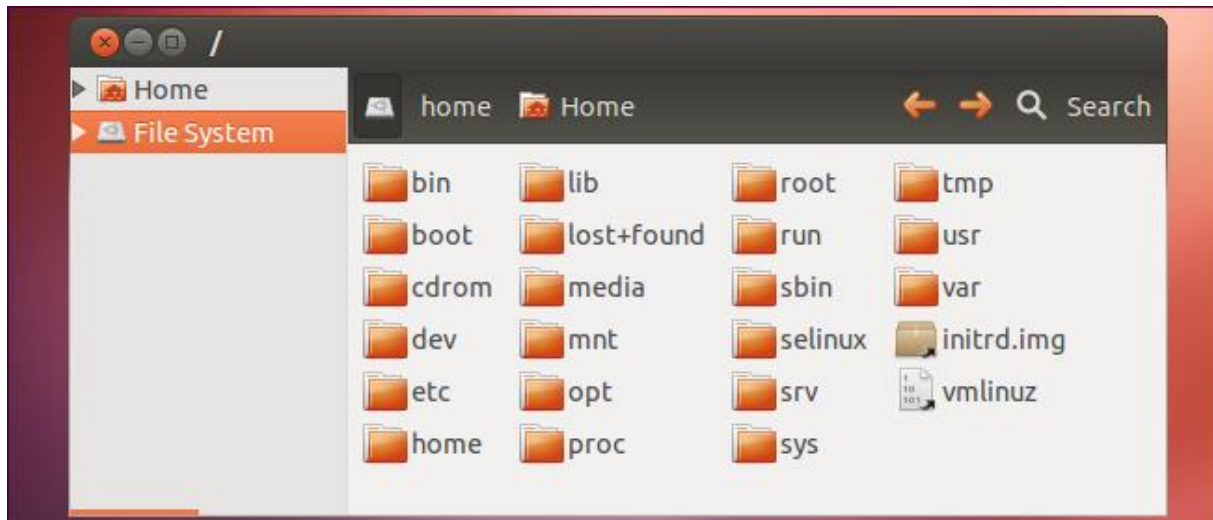
Unterlagen:

- [Boot Prozess](#)
- [System V](#)

3.4.3.1 Where are Programs and Configurations stored ?

Hier eine gute Übersicht, wo die jeweiligen Dateien gespeichert sind:



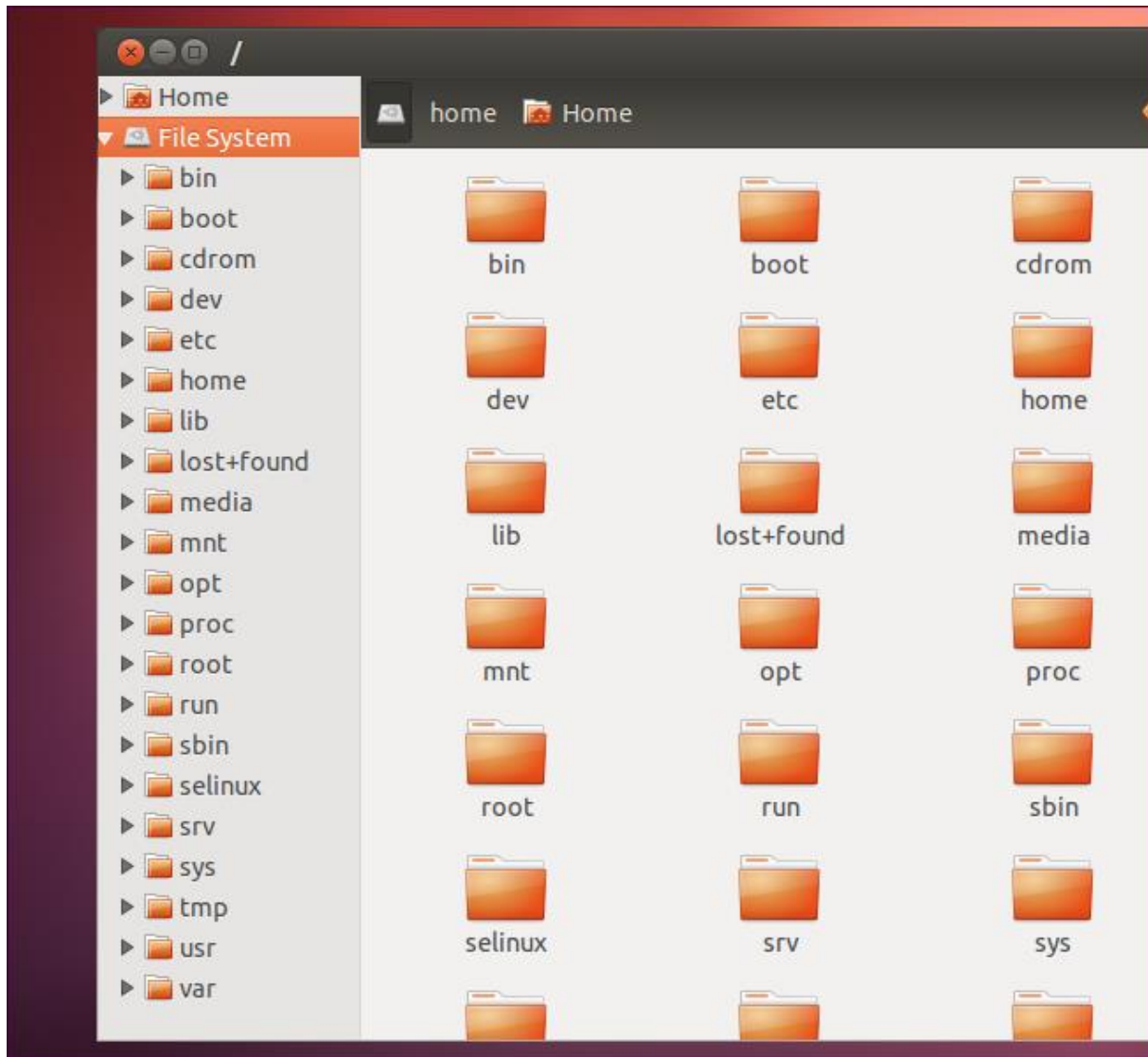


If you're coming from Windows, the Linux file system structure can seem particularly alien. The C:\ drive and drive letters are gone, replaced by a / and cryptic-sounding directories, most of which have three letter names.

The Filesystem Hierarchy Standard (FHS) defines the structure of file systems on Linux and other UNIX-like operating systems. However, Linux file systems also contain some directories that aren't yet defined by the standard.

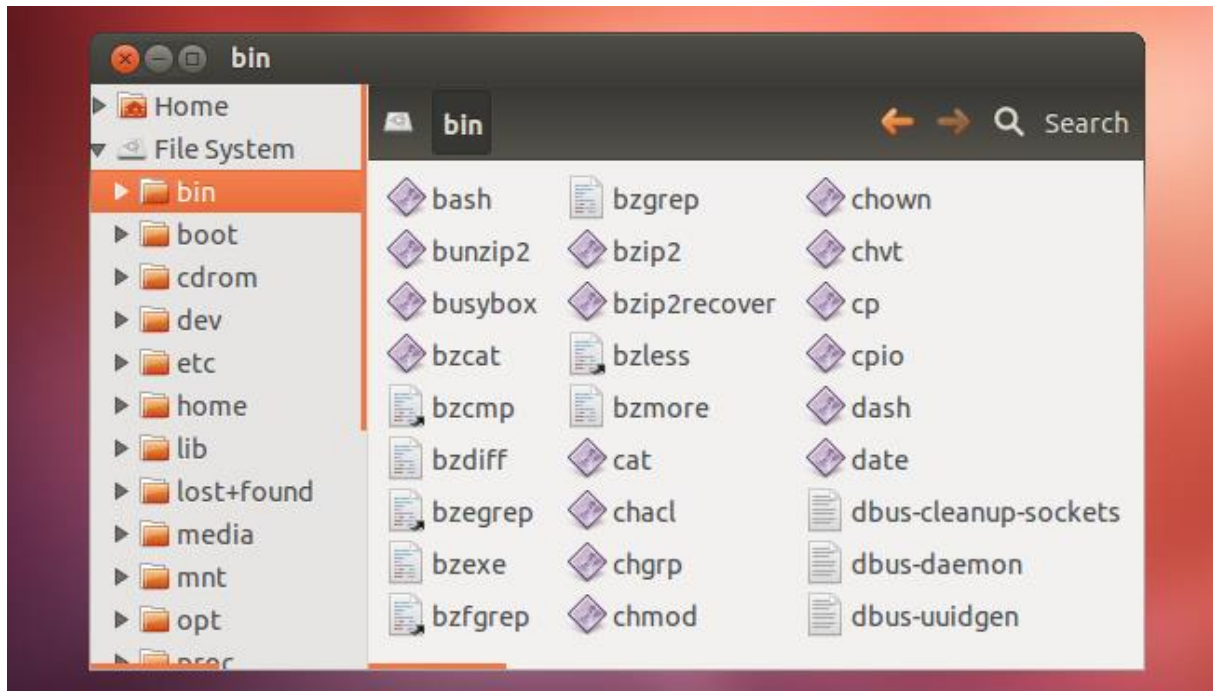
3.4.3.1.1 / – The Root Directory

Everything on your Linux system is located under the / directory, known as the root directory. You can think of the / directory as being similar to the C:\ directory on Windows – but this isn't strictly true, as Linux doesn't have drive letters. While another partition would be located at D:\ on Windows, this other partition would appear in another folder under / on Linux.



3.4.3.1.2 /bin – Essential User Binaries

The /bin directory contains the essential user binaries (programs) that must be present when the system is mounted in single-user mode. Applications such as Firefox are stored in /usr/bin, while important system programs and utilities such as the bash shell are located in /bin. The /usr directory may be stored on another partition – placing these files in the /bin directory ensures the system will have these important utilities even if no other file systems are mounted. The /sbin directory is similar – it contains essential system administration binaries.



3.4.3.1.3 /boot – Static Boot Files

The `/boot` directory contains the files needed to boot the system – for example, the GRUB boot loader’s files and your Linux kernels are stored here. The boot loader’s configuration files aren’t located here, though – they’re in `/etc` with the other configuration files.

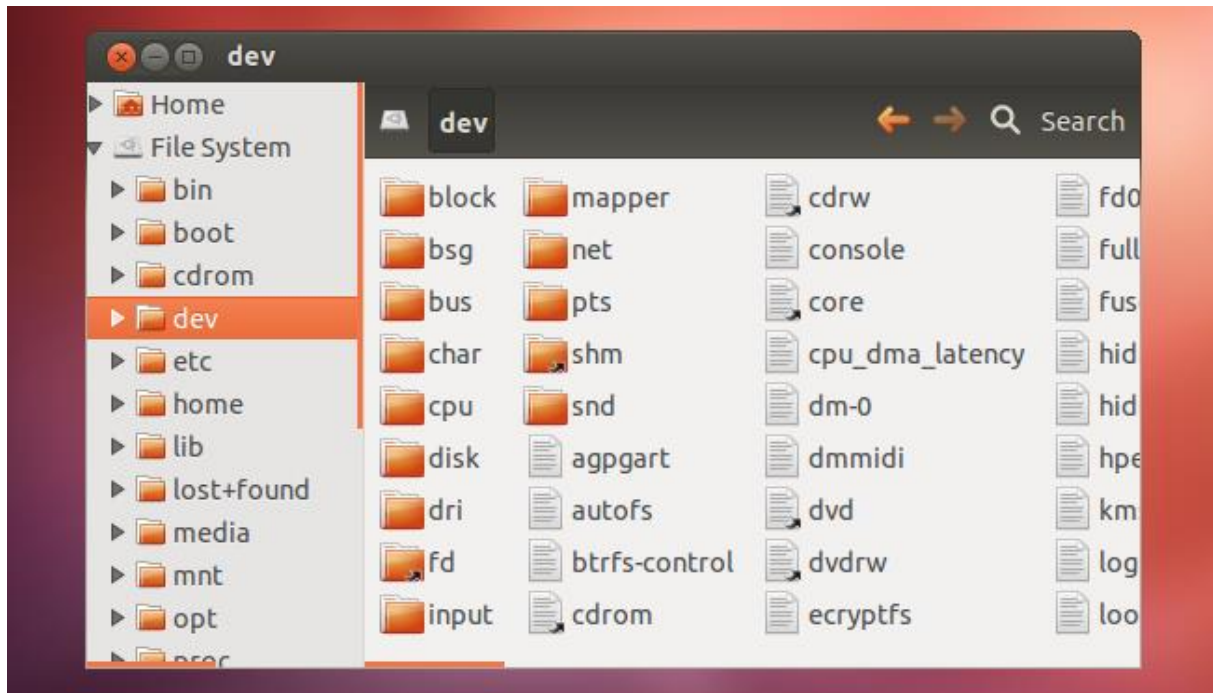
3.4.3.1.4 /cdrom – Historical Mount Point for CD-ROMs

The /cdrom directory isn't part of the FHS standard, but you'll still find it on Ubuntu and other operating systems. It's a temporary location for CD-ROMs inserted in the system. However, the standard location for temporary media is inside the /media directory.

3.4.3.1.5 /dev – Device Files

Linux exposes devices as files, and the `/dev` directory contains a number of special files that represent devices. These are not actual files as we know them, but they appear as files – for example, `/dev/sda` represents the first SATA drive in the system. If you wanted to partition it, you could start a partition editor and tell it to edit `/dev/sda`.

This directory also contains pseudo-devices, which are virtual devices that don't actually correspond to hardware. For example, `/dev/random` produces random numbers. `/dev/null` is a special device that produces no output and automatically discards all input – when you pipe the output of a command to `/dev/null`, you discard it.

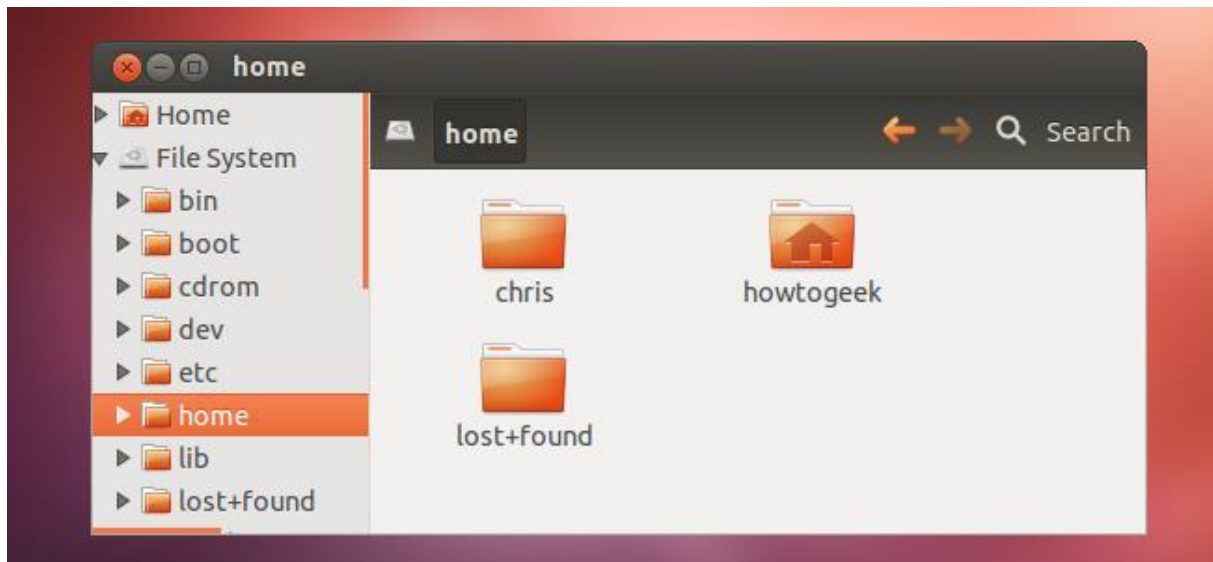


3.4.3.1.6 /etc – Configuration Files

The /etc directory contains configuration files, which can generally be edited by hand in a text editor. Note that the /etc/ directory contains system-wide configuration files – user-specific configuration files are located in each user's home directory.

3.4.3.1.7 /home – Home Folders

The /home directory contains a home folder for each user. For example, if your user name is bob, you have a home folder located at /home/bob. This home folder contains the user's data files and user-specific configuration files. Each user only has write access to their own home folder and must obtain elevated permissions (become the root user) to modify other files on the system.



3.4.3.1.8 /lib – Essential Shared Libraries

The /lib directory contains libraries needed by the essential binaries in the /bin and /sbin folder. Libraries needed by the binaries in the /usr/bin folder are located in /usr/lib.

3.4.3.1.9 /lost+found – Recovered Files

Each Linux file system has a lost+found directory. If the file system crashes, a file system check will be performed at next boot. Any corrupted files found will be placed in the lost+found directory, so you can attempt to recover as much data as possible.

3.4.3.1.10 /media – Removable Media

The /media directory contains subdirectories where removable media devices inserted into the computer are mounted. For example, when you insert a CD into your Linux system, a directory will automatically be created inside the /media directory. You can access the contents of the CD inside this directory.

3.4.3.1.11 /mnt – Temporary Mount Points

Historically speaking, the /mnt directory is where system administrators mounted temporary file systems while using them. For example, if you're mounting a Windows

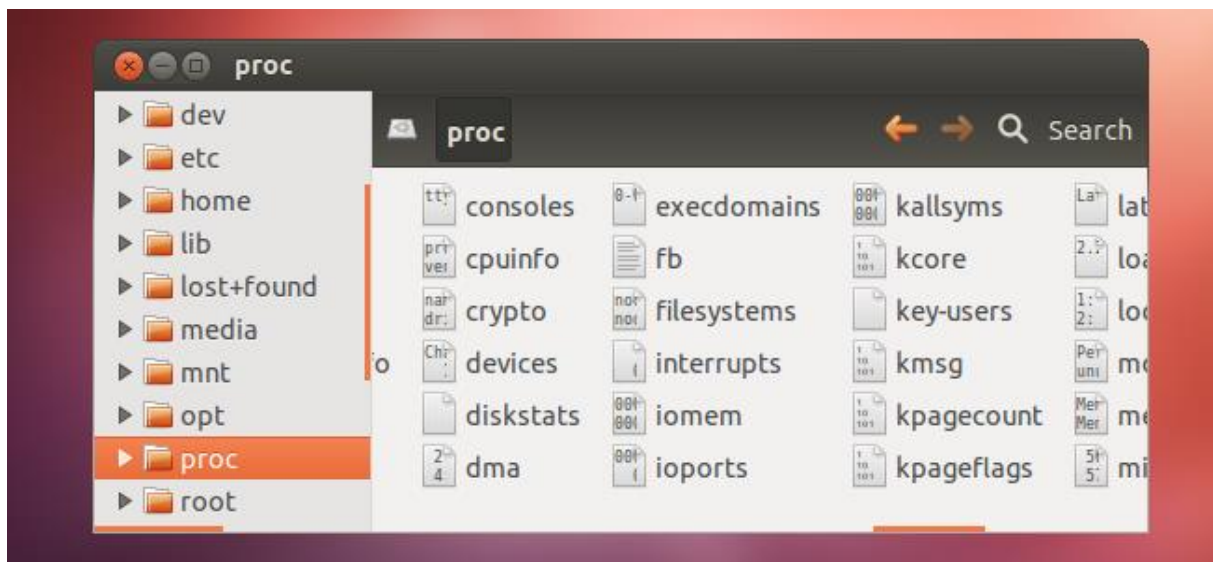
partition to perform some file recovery operations, you might mount it at /mnt/windows. However, you can mount other file systems anywhere on the system.

3.4.3.1.12 /opt – Optional Packages

The /opt directory contains subdirectories for optional software packages. It's commonly used by proprietary software that doesn't obey the standard file system hierarchy – for example, a proprietary program might dump its files in /opt/application when you install it.

3.4.3.1.13 /proc – Kernel & Process Files

The /proc directory is similar to the /dev directory because it doesn't contain standard files. It contains special files that represent system and process information.



3.4.3.1.14 /root – Root Home Directory

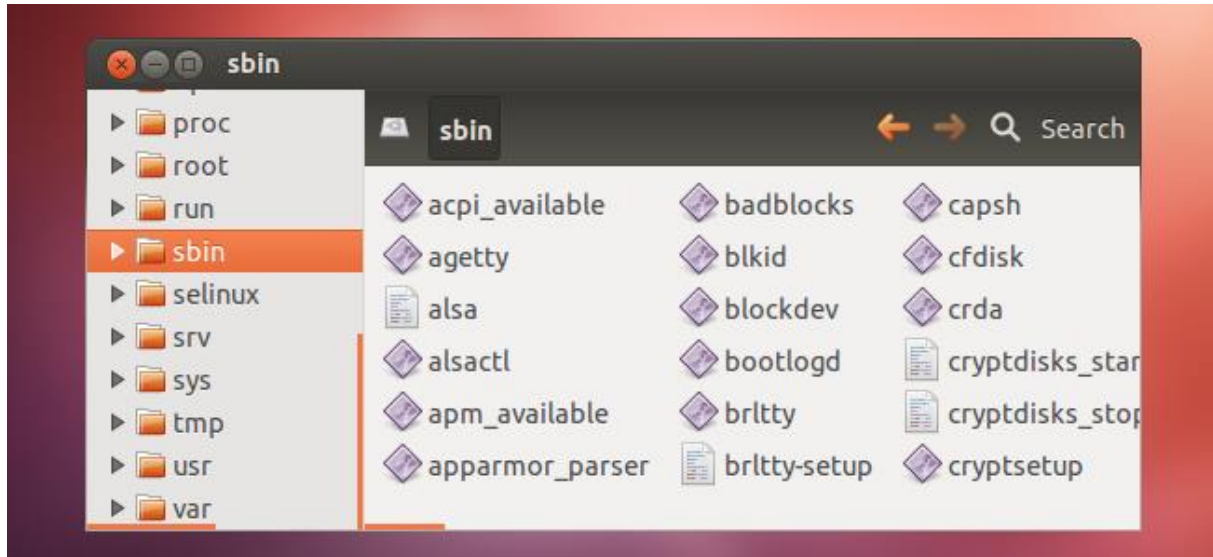
The /root directory is the home directory of the root user. Instead of being located at /home/root, it's located at /root. This is distinct from /, which is the system root directory.

3.4.3.1.15 /run – Application State Files

The /run directory is fairly new, and gives applications a standard place to store transient files they require like sockets and process IDs. These files can't be stored in /tmp because files in /tmp may be deleted.

3.4.3.1.16 /sbin – System Administration Binaries

The /sbin directory is similar to the /bin directory. It contains essential binaries that are generally intended to be run by the root user for system administration.



3.4.3.1.17 /selinux – SELinux Virtual File System

If your Linux distribution uses SELinux for security (Fedora and Red Hat, for example), the /selinux directory contains special files used by SELinux. It's similar to /proc. Ubuntu doesn't use SELinux, so the presence of this folder on Ubuntu appears to be a bug.

3.4.3.1.18 /srv – Service Data

The /srv directory contains “data for services provided by the system.” If you were using the Apache HTTP server to serve a website, you'd likely store your website's files in a directory inside the /srv directory.

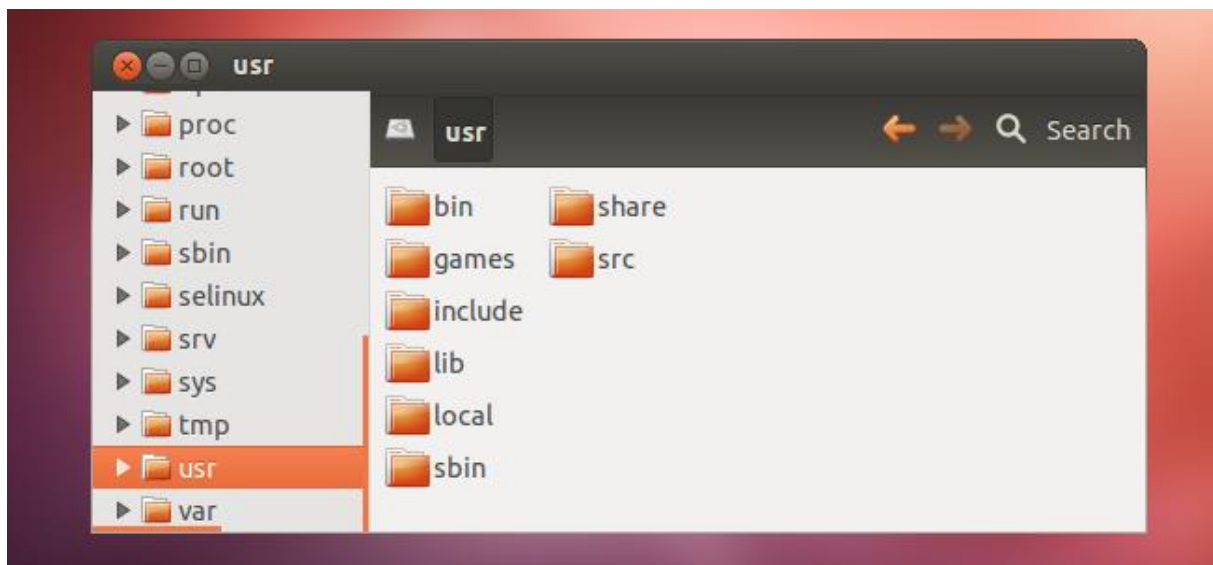
3.4.3.1.19 /tmp – Temporary Files

Applications store temporary files in the /tmp directory. These files are generally deleted whenever your system is restarted and may be deleted at any time by utilities such as tmpwatch.

3.4.3.1.20 /usr – User Binaries & Read-Only Data

The /usr directory contains applications and files used by users, as opposed to applications and files used by the system. For example, non-essential applications are located inside the /usr/bin directory instead of the /bin directory and non-essential system administration binaries are located in the /usr/sbin directory instead of the /sbin directory. Libraries for each are located inside the /usr/lib directory. The /usr directory also contains other directories – for example, architecture-independent files like graphics are located in /usr/share.

The /usr/local directory is where locally compiled applications install to by default – this prevents them from mucking up the rest of the system.



3.4.3.1.21 /var – Variable Data Files

The /var directory is the writable counterpart to the /usr directory, which must be read-only in normal operation. Log files and everything else that would normally be written to /usr during normal operation are written to the /var directory. For example, you'll find log files in /var/log.

Quelle: <https://www.howtogeek.com/117435/htg-explains-the-linux-directory-structure-explained/>

3.4.4 Your Computer on the Network

Weight: 2

Description: Querying vital networking configuration and determining the basic requirements for a computer on a Local Area Network (LAN).

Key Knowledge Areas:

- Internet, network, routers
- Querying DNS client configuration
- Querying network configuration

The following is a partial list of the used files, terms and utilities:

- route, ip route show
- ifconfig, ip addr show
- netstat, ss
- /etc/resolv.conf, /etc/hosts
- IPv4, IPv6
- ping
- host

Unterlagen:

- [Linux Essentials](#) von [tuxcademy](#) - Kapitel 15
- [Standard Datenströme](#)
- [File Manipulationen](#)

4. Kubernetes Installieren

4.1 Pre-Installation Steps On Both Master & Slave (To Install Kubernetes)

The following steps have to be executed on both the master and node machines. Let's call the master as '*kmaster*' and node as '*knode*'.

First, login as 'sudo' user because the following set of commands need to be executed with 'sudo' permissions. Then, update your 'apt-get' repository.

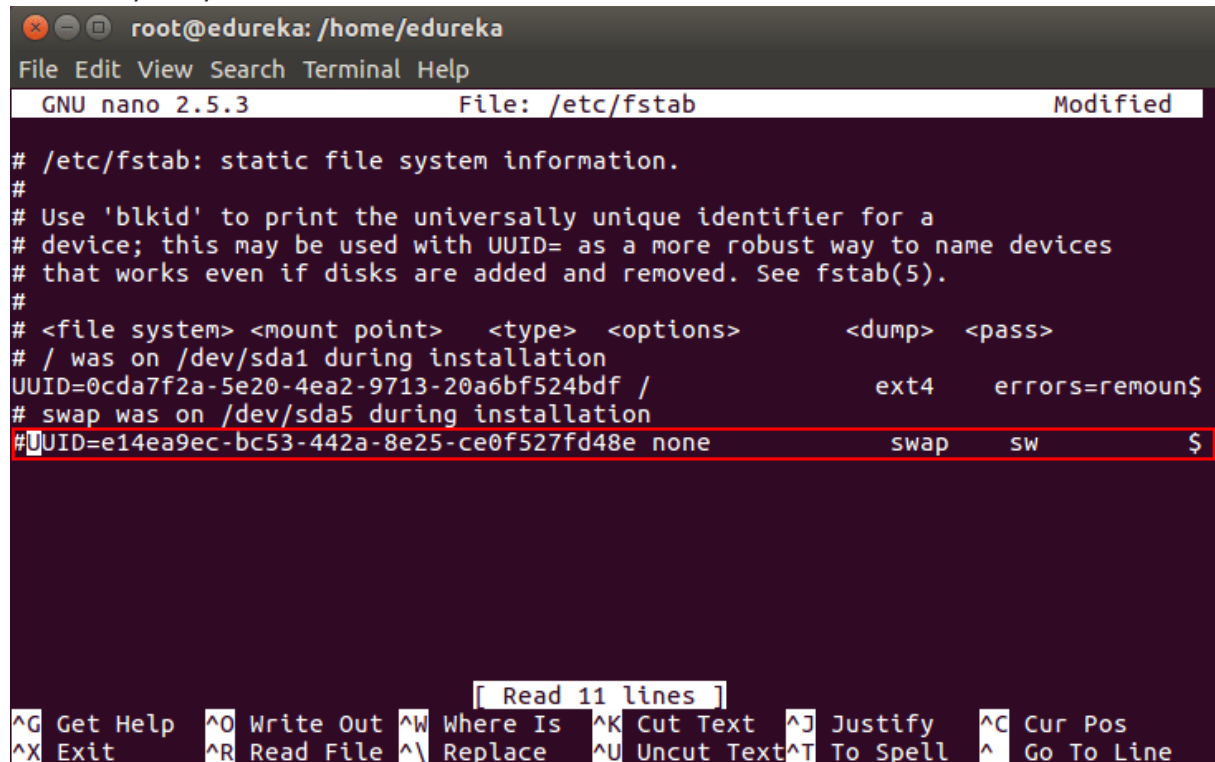
```
$ sudo su
# apt-get update
```

Note: After logging-in as 'sudo' user, note that your shell symbol will change to '#' from '\$'.

4.2 Turn Off Swap Space

Next, we have to turn off the swap space because Kubernetes will start throwing random errors otherwise. After that you need to open the 'fstab' file and comment out the line which has mention of swap partition.

```
# swapoff -a
# nano /etc/fstab
```



```

root@edureka: /home/edureka
File Edit View Search Terminal Help
GNU nano 2.5.3 File: /etc/fstab Modified

# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options> <dump> <pass>
# / was on /dev/sda1 during installation
UUID=0cda7f2a-5e20-4ea2-9713-20a6bf524bdf / ext4 errors=remoun$
# swap was on /dev/sda5 during installation
#UUID=e14ea9ec-bc53-442a-8e25-ce0f527fd48e none swap sw $

[ Read 11 lines ]
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line

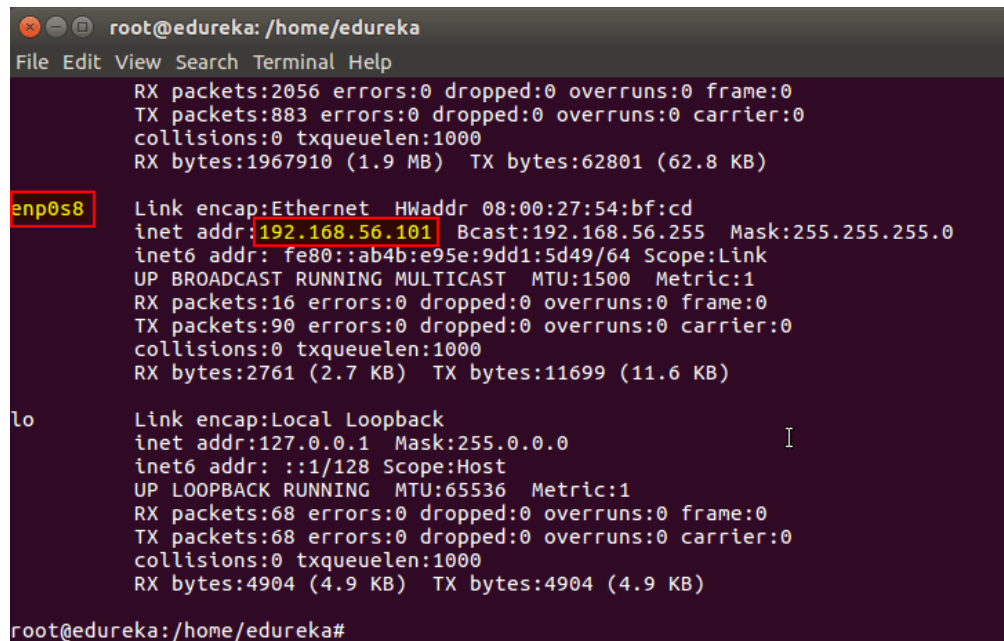
```

4.3 Update The Hosts File With IPs Of Master & Node

Run the following command on both machines to note the IP addresses of each.

ifconfig

Make a note of the IP address from the output of the above command. The IP address which has to be copied should be under “enp0s8”, as shown in the screenshot below.



```

root@edureka: /home/edureka
File Edit View Search Terminal Help
RX packets:2056 errors:0 dropped:0 overruns:0 frame:0
TX packets:883 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:1967910 (1.9 MB) TX bytes:62801 (62.8 KB)

enp0s8: Link encap:Ethernet HWaddr 08:00:27:54:bf:cd
        inet addr:192.168.56.101 Bcast:192.168.56.255 Mask:255.255.255.0
        inet6 addr: fe80::ab4b:e95e:9dd1:5d49/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        RX packets:16 errors:0 dropped:0 overruns:0 frame:0
        TX packets:90 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:2761 (2.7 KB) TX bytes:11699 (11.6 KB)

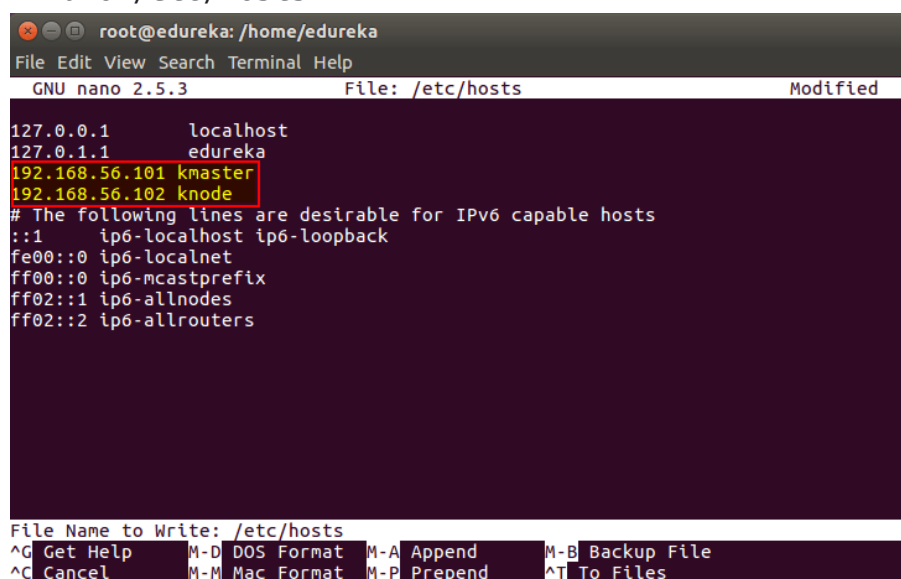
lo: Link encap:Local Loopback
        inet addr:127.0.0.1 Mask:255.0.0.0
        inet6 addr: ::1/128 Scope:Host
        UP LOOPBACK RUNNING MTU:65536 Metric:1
        RX packets:68 errors:0 dropped:0 overruns:0 frame:0
        TX packets:68 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:4904 (4.9 KB) TX bytes:4904 (4.9 KB)

root@edureka: /home/edureka#

```

Now go to the ‘hosts’ file on both the master and node and add an entry specifying their respective IP addresses along with their names ‘kmaster’ and ‘knode’. This is used for referencing them in the cluster. It should look like the below screenshot on both the machines.

nano /etc/hosts



```

root@edureka: /home/edureka
File Edit View Search Terminal Help
GNU nano 2.5.3 File: /etc/hosts Modified

127.0.0.1    localhost
127.0.1.1    edureka
192.168.56.101 kmaster
192.168.56.102 knode
# The following lines are desirable for IPv6 capable hosts
::1         ip6-localhost ip6-loopback
fe00::0     ip6-localnet
ff00::0     ip6-mcastprefix
ff02::1     ip6-allnodes
ff02::2     ip6-allrouters

File Name to Write: /etc/hosts
^G Get Help      M-D DOS Format  M-A Append     M-B Backup File
^C Cancel        M-M Mac Format  M-P Prepend    ^T To Files

```

4.4 Setting Static IP Addresses

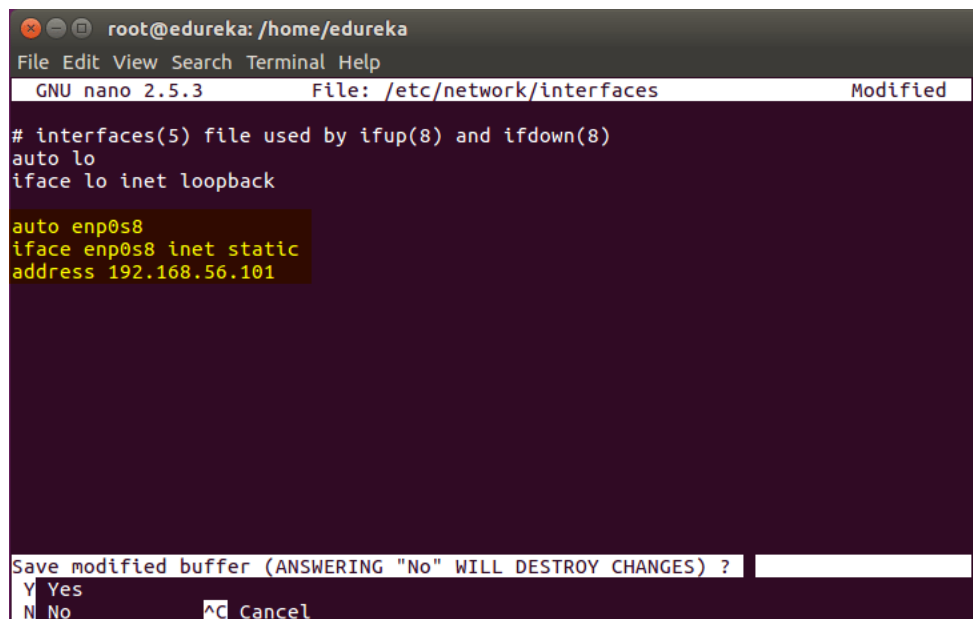
Next, we will make the IP addresses used above, static for the VMs. We can do that by modifying the network interfaces file. Run the following command to open the file:

```
# nano /etc/network/interfaces
```

Now enter the following lines in the file.

```
auto enp0s8
iface enp0s8 inet static
address <IP-Address-Of-VM>
```

It will look something like the below screenshot.



```
root@edureka: /home/edureka
File Edit View Search Terminal Help
GNU nano 2.5.3 File: /etc/network/interfaces Modified

# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

auto enp0s8
iface enp0s8 inet static
address 192.168.56.101

Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES) ?
Y Yes
N No ^C Cancel
```

After this, restart your machine(s).

4.5.1 Install OpenSSH-Server

Now we have to install openssh-server. Run the following command:

```
# sudo apt-get install openssh-server
```

4.5.2 Install Docker

Now we have to install Docker because Docker images will be used for managing the containers in the cluster. Run the following commands:

```
# sudo su
# apt-get update
# apt-get install -y docker.io
```

Next we have to install these 3 essential components for setting up Kubernetes environment: kubeadm, kubectl, and kubelet.

Run the following commands before installing the Kubernetes environment.

```
# apt-get update && apt-get install -y apt-transport-https curl
# curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg |
apt-key add -
# cat <<EOF >/etc/apt/sources.list.d/kubernetes.list
deb http://apt.kubernetes.io/ kubernetes-xenial main
EOF
# apt-get update
```

4.5.3 Install kubeadm, Kubelet And Kubectl

Now its time to install the 3 essential components. **Kubelet** is the lowest level component in Kubernetes. It's responsible for what's running on an individual machine. **Kubeadm** is used for administrating the Kubernetes cluster. **Kubectl** is used for controlling the configurations on various nodes inside the cluster.

```
# apt-get install -y kubelet kubeadm kubectl
```

4.5.4 Updating Kubernetes Configuration

Next, we will change the configuration file of Kubernetes. Run the following command:

```
# nano /etc/systemd/system/kubelet.service.d/10-kubeadm.conf
```

This will open a text editor, enter the following line after the last "Environment Variable":

```
Environment="cgroup-driver=systemd/cgroup-driver=cgroupfs"
```

```
root@kmaster: /home/edureka
GNU nano 2.5.3 File: ...md/system/kubelet.service.d/10-kubeadm.conf Modified
[Service]
Environment="KUBELET_KUBECONFIG_ARGS=--bootstrap-kubeconfig=/etc/kubernetes/boo$
Environment="KUBELET_SYSTEM_PODS_ARGS=--pod-manifest-path=/etc/kubernetes/manif$
Environment="KUBELET_NETWORK_ARGS=--network-plugin=cni --cni-conf-dir=/etc/cni/$
Environment="KUBELET_DNS_ARGS=--cluster-dns=10.96.0.10 --cluster-domain=cluster$
Environment="KUBELET_AUTHZ_ARGS=--authorization-mode=Webhook --client-ca-file=$
Environment="KUBELET_CADVISOR_ARGS=--cadvisor-port=0"
Environment="KUBELET_CERTIFICATE_ARGS=--rotate-certificates=true --cert-dir=/va$
Environment="cgroup-driver=systemd/cgroup-driver=cgroupfs"
ExecStart=
ExecStart=/usr/bin/kubelet $KUBELET_KUBECONFIG_ARGS $KUBELET_SYSTEM_PODS_ARGS $$

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^I Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

Voila! You have successfully installed Kubernetes on both the machines now!

4.6 Steps Only For Kubernetes Master VM (kmaster)

Note: These steps will only be executed on the master node (kmaster VM).

Step 1: We will now start our Kubernetes cluster from the master's machine. Run the following command:

```
# sudo kubeadm init --apiserver-advertise-address=<ip-address-of-
kmaster-vm> --pod-network-cidr=192.168.0.0/16
```

1. You will get the below output. The commands marked as (1), execute them as a non-root user. This will enable you to use kubectl from the CLI
2. The command marked as (2) should also be saved for future. This will be used to join nodes to your cluster

```
root@kmaster: /home/edureka
space
[addons] Applied essential addon: kube-dns
[addons] Applied essential addon: kube-proxy

Your Kubernetes master has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

You can now join any number of machines by running the following on each node
as root:

  kubeadm join 192.168.56.101:6443 --token k44k0v.u2s9q6gjoykpoxk0 --discovery-t
oken-ca-cert-hash sha256:d210bd373c0c9d628260496c90b23f62c3c8e89f0a41f26f223fed6
8a30e31ba

root@kmaster: /home/edureka#
```

Step 2: As mentioned before, run the commands from the above output as a non-root user

```
$ mkdir -p $HOME/.kube
$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
```


It should look like this:

```

edureka@kmaster: ~
edureka@kmaster:~$ mkdir -p $HOME/.kube
edureka@kmaster:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
edureka@kmaster:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
edureka@kmaster:~$

```

To verify, if kubectl is working or not, run the following command:

```
$ kubectl get pods -o wide --all-namespaces
```

```

edureka@kmaster: ~
edureka@kmaster:~$ kubectl get pods -o wide --all-namespaces
NAMESPACE   NAME                                READY   STATUS    RESTARTS   AGE   IP            NODE
kube-system  etcd-kmaster                        1/1     Running   0           4s    192.168.56.101  kmaster
kube-system  kube-apiserver-kmaster              1/1     Running   0           4s    192.168.56.101  kmaster
kube-system  kube-controller-manager-kmaster     1/1     Running   0           4s    192.168.56.101  kmaster
kube-system  kube-dns-86f4d74b45-ggg8z          0/3     Pending   0           12m    <none>          <none>
kube-system  kube-proxy-85tp2                    1/1     Running   0           12m    192.168.56.101  kmaster
kube-system  kube-scheduler-kmaster              1/1     Running   0           4s    192.168.56.101  kmaster
edureka@kmaster:~$

```

Step 3: You will notice from the previous command, that all the pods are running except one: 'kube-dns'. For resolving this we will install a pod network. To install the CALICO pod network, run the following command:

```

Sudo kubectl apply -f https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/hosted/rbac-kdd.yaml
Sudo kubectl apply -f https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/hosted/kubernetes-datastore/calico-networking/1.7/calico.yaml

```

After some time, you will notice that all pods shift to the running state.

Step 4: Next, we will install the dashboard. To install the Dashboard, run the following command:

```
Sudo kubectl apply -f
https://raw.githubusercontent.com/kubernetes/dashboard/master/aio/deploy/recommended/kubernetes-dashboard.yaml
```

Step 5: Your dashboard is now ready with it's the pod in the running state.

kube-system	etcd-kmaster	1/1	Running	0
kube-system	kube-apiserver-kmaster	1/1	Running	0
kube-system	kube-controller-manager-kmaster	1/1	Running	0
kube-system	kube-dns-86f4d74b45-ggg8z	3/3	Running	0
kube-system	kube-proxy-85tp2	1/1	Running	0
kube-system	kube-scheduler-kmaster	1/1	Running	0
kube-system	kubernetes-dashboard-7d5dcdb6d9-bbmmr	1/1	Running	0

Step 6: By default dashboard will not be visible on the Master VM. Run the following command in the command line:

```
$ kubectl proxy
```

To view the dashboard in the browser, navigate to the following address in the browser of your Master VM:

<http://localhost:8001/api/v1/namespaces/kube-system/services/https:kubernetes-dashboard:/proxy/>

You will then be prompted with this page, to enter the credentials:

Kubernetes Dashboard

☒ Kubeconfig

Please select the kubeconfig file that you have created to configure access to the cluster. To find out more about how to configure and use kubeconfig file, please refer to the [Configure Access to Multiple Clusters](#) section.

☐ Token

Every Service Account has a Secret with valid Bearer Token that can be used to log in to Dashboard. To find out more about how to configure and use Bearer Tokens, please refer to the [Authentication](#) section.

SIGN IN

SKIP

Step 7: In this step, we will create the service account for the dashboard and get it's credentials.

Note: Run all these commands in a new terminal, or your kubectl proxy command will stop.

Run the following commands:

1. This command will create a service account for dashboard in the default namespace

```
$ kubectl create serviceaccount dashboard -n default
```

2. This command will add the cluster binding rules to your dashboard account

```
$ kubectl create clusterrolebinding dashboard-admin -n default --clusterrole=cluster-admin --serviceaccount=default:dashboard
```

3. This command will give you the token required for your dashboard login:

```
kubectl get secret $(kubectl get serviceaccount dashboard -o
jsonpath="{.secrets[0].name}") -o jsonpath="{.data.token}" | base64
--decode
```

You should get the token like this:

```
edureka@kmaster:~$ kubectl get secret $(kubectl get serviceaccount dashboard -o jsonpath="{.secrets[0].name}") -o jsonpath="{.data.token}" | base64 --decode
eyJhbGciOiJIUzI1NiIsImtpZCI6IiJ9.eyJpc3MiOiJrdWJlcm5ldGVzZ3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWVhY2NvdW50L3NlcnZpY2VhY2NvdW50L21hY2N0eSIsImt1YmVybW0ZXMuaW8vc2VydmJpZWFjY291bnQvc2VydmljZS1hY2NvdW50Lm5hbWU0IjJkYXNoYm9mcmQlLCJlcnRlcm5ldGVzLmVlL3NlcnZpY2VhY2NvdW50L3NlcnZpY2VhY2NvdW50L1h0b3VudC51aWQ0IjJhYWY1YzI1MS01YWE0LExZTgtOGY3YS0wODAwMjdmODRlZkZlZmVhY2N0ZW06c2VydmJpZWFjY291bnQ6ZGVmYXVsdDpkYXNoYm9mcmQifQ.wKPklojENDmJ4l74LhQNCHTQ2Gs2julo0vYdk4pkU4vN8iB54x7I9Bq0YUiuJw_zEZqjnWyQdjdDu2DAMtXwC_5uILo4SaTtL_bVaRVrb0oVCxxELaUyHQfppzEL8-EJNXXGUuIqzvzYr8zkYRTaQ1cjb3tXBlCrG5Ru-moN7IdPwXxaerWjdJWiH96h_VRM05myiCoX_gTBHztWQ00sdgOWUff2fTodCo-e516vxBzN0THKdGKBE2m7FenwXCCLTkZwHUHUK6yZuJq_vDpOn1P7ARqQYnwXjh6eHzKqJ9b8rf41D6m6DmLS0vgd0SCPfwijkz_ppv_tl-XVPdT0
```

- Copy this token and paste it in Dashboard Login Page, by selecting token option

Kubernetes Dashboard

☐ Kubeconfig

Please select the kubeconfig file that you have created to configure access to the cluster. To find out more about how to configure and use kubeconfig file, please refer to the [Configure Access to Multiple Clusters](#) section.

☒ Token

Every Service Account has a Secret with valid Bearer Token that can be used to log in to Dashboard. To find out more about how to configure and use Bearer Tokens, please refer to the [Authentication](#) section.

Enter token

.....

SIGN IN SKIP

- You have successfully logged into your dashboard!

The screenshot shows the Kubernetes Dashboard interface. The top navigation bar includes the Kubernetes logo, the word "kubernetes", and a search bar. The left sidebar shows the "Overview" menu. The main content area is divided into sections: "Cluster" (with links to Namespaces, Nodes, Persistent Volumes, Roles, and Storage Classes), "Discovery and Load Balancing" (containing a "Services" table), and "Config and Storage".

Name	Labels	Cluster IP	Internal endpoints
✓ kubernetes	component: apiserver provider: kubernetes	10.96.0.1	kubernetes:443 TCP kubernetes:0 TCP

4.7 Steps For Only Kubernetes Node VM (knode)

It is time to get your node, to join the cluster! This is probably the only step that you will be doing on the node, after installing kubernetes on it.

Run the join command that you saved, when you ran 'kubeadm init' command on the master.

Note: Run this command with "sudo".

```
sudo kubeadm join --apiserver-advertise-address=<ip-address-of-the master> --pod-network-cidr=192.168.0.0/16
```

```
root@knode:/home/user# kubeadm join 192.168.119.132:6443 --token acse04.fljmb1mco8s6d0rw --discovery-token-ca-cert-hash sha256:ad8a501aaabe273b44e83e203bcf539d747865136ed8ee087ffa0f1bdb2404e6
[preflight] Running pre-flight checks
[WARNING IsDockerSystemdCheck]: detected "cgroupfs" as the Docker cgroup driver. The recommended driver is "systemd". Please follow the guide at https://kubernetes.io/docs/setup/cri/
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -oyaml'
[kubelet-start] Downloading configuration for the kubelet from the "kubelet-config-1.14" ConfigMap in the kube-system namespace
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Activating the kubelet service
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
```

Bingo! Your Kubernetes Cluster is ready if you get something similar to the above screenshot.

Source: <https://www.edureka.co/blog/install-kubernetes-on-ubuntu#StartingKubernetesCluster>

5 Hinzufügen eines Nginx Servers in Kubernetes

How-To: <https://kubernetes.io/docs/tasks/run-application/run-stateless-application-deployment/>
<https://github.com/mc-b/M300/blob/master/40-Kubernetes/K8s.md>

Nachdem wir die Pods erzeugt haben, müssen wir den Service erzeugen. Dadurch wird der Web Server von aussen sichtbar:

```
kubect1 expose deployments/nginx-deployment --type="LoadBalancer" --port 80
```