FullDiscEncryption (Festplattenverschlüsselung) auf dem eigenen Server

## Inhaltsverzeichnis



# 1. Abstrakt / Vorwort

Dieses Artikel ist Teil einer sehr losen Artikelreihe, die ich auf http://lug-hamburg.de posten werde.

Im diesem Paper beschreibe ich die Neu-Installation einer Debian-Basis-Installation mit der Einrichtung einer Festplattenvollverschlüsselung auf dem eigenen Server, um diesen vor Fremdzugriff zu schützen. Mit leichten Anpassungen ist dieser Artikel auch für die Festplattenvollverschlüsselung des eigenen Notebooks verwendbar.

Schwierigkeitsgrad:  mittel. Eine gewisse Affinität zu Linux und Cryptothemen ist sicher hilfreich, ebenso wie Geduld beim Lesen dieses Artikels. Ich versuche das Thema jedoch möglichst ausführlich zu behandeln, kann aber aus sicher nachvollziehbaren Gründen nicht bei „Adam und Eva“ anfangen.

Die Beschreibung erfolgt am Beispiel von Debian (ich verwende für die Beispielinstallation Debian 8 Jessie und Stretch). Praktisch nutze ich diese Anleitung auf dedizierten OVH-Server (dort: Kimsufi-Reihe, https://www.kimsufi.com/en/ ) und Online.net-Servern.
Wichtig zu erwähnen ist, dass ich hier auf die Arbeit anderer aufbaue, denen ich sehr für die Vorarbeiten danke. Links dazu folgen im nächsten Kapitel.

Alle in diesem Dokument vorgestellten Anweisungen und Befehle sind nach bestem Wissen zusammengestellt, für die Richtigkeit übernehme ich jedoch keine Haftung!

Dieses stellt meine Herangehensweise dar und erhebt keinen Anspruch auf eine perfekte, optimale oder gar funktionsfähige Installation. Daher:
bitte gerne bei Fehlern/Unvollständigkeiten melden.

Januar 2015, Karsten Wiborg

# 2. Links

- Grundlagenartikel von Martín Ferrari:
  - http://blog.tincho.org/posts/Setting_up_my_server:_re-installing_on_an_encripted_LVM/
- Infos zu FullDisc-Encryption mit Online.net:
  - https://github.com/Val/dedibox_fully_encrypted_debian_install
- Man-Pages für cryptsetup, LVM, etc.

# 3. Versionierung

| Version | Änderung am und durch | Änderungen / Kommentare |
|---|---|---|
| 0.1 | 31.12.2014 / Karsten Wiborg | Ersterstellung, Lizenz: CC BY-SA 4.0 |
| 0.2 | 03.05.2015 / Karsten Wiborg | - neues Manager-Interface dokumentiert<br>- Ergänzungen zu Debian 8 „Jessie“ |
| 0.3 | 22.01.2017 / Karsten Wiborg | - Anbieter online.net als Einrichtungsvariante hinzugefügt<br>- Ergänzungen zu Debian 9 „Stretch“<br>- Wheezy entfernt |

# 4. Voraussetzungen

- Ein Server mit einer Rescue-Konsole (z.B. die Rescue-Konsole bei OVH bzw. Online.net oder aber auch ein iLo-/DRAC-Interface mit der Möglichkeit remote ein ISO-Image zu mounten) und vollen Zugriff darauf. Dieses Beispiel bezieht sich auf die OVH-/Online.net-Rescue-Konsole.
- Zugriff auf die folgenden erweiterten Befehle in der Rescue-Konsole:
  - chroot
  - cryptsetup
  - die gängigen lvm-Befehle
  - debootstrap
- eine vorbereitete authorized_keys-Datei mit dem eigenen SSH-Public-Key (wer sich hiermit nicht auskennt: bitte sich erst einmal mit dem Thema SSH Public-/Private-Keying auseinandersetzen).

# 5. Umsetzung

**ACHTUNG:**

- bei Debian Jessie klappte diese Beschreibung anfangs eingeschränkt mit einer frischen Installation während ein Update von Wheezy auf Jessie bei mir einwandfrei klappte! Sollte es Ärger mit Jessie geben: das Aufschliessen des Crypto-Volumes klappte mit den ersten Jessie-Releases nicht mehr via /lib/cryptsetup/passfifo, da unter /lib kein cryptsetup mehr existierte! Einen Workaround zum Entsperren findet der geneigte Leser unter „Entsperren des Cryptocontainers beim Booten“. Debian 9 „Stretch“ hat dagegen bei mir bis jetzt (Stand 01/2017) einwandfrei geklappt.

    - Link: https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=783631
- Seit Debian 9 „Stretch“ gibt es nicht mehr die üblichen eth0-Ethernet-Interfaces. Stattdessen sind die Interfaces jetzt hardwarenah bezeichnet (z.B. enp1s0). Das bringt das Problem mit sich, dass man entweder vorher eine Standardinstallation vom Anbieter durchführen lässt (sofern dieser bereits Debian 9 unterstützt) oder man sich den Interfacenamen vorher berechnet. Das sollte dann aber passen, da sonst der Server mit dem falschen Interfacenamen keine Netzwerkkonfiguration ziehen kann. Näheres zur Vorberechnung des Netzwerkkarten namens weiter unten.

Noch einmal der Hinweis:

Die folgende Anleitung ist sehr ausführlich und und installiert den kompletten Server mit Festplattenverschlüsselung neu. **Es gehen also alle Daten verloren**, die auf dem Server möglicherweise noch liegen. Da es sich hier um eine komplexere Installation handelt, sollte möglichst genau den Hinweisen folge geleistet werden. Wobei auch hier gilt: ich hoffe, ich habe keine Fehler in den Text einfließen lassen. Allerdings habe ich bereits mehrere Server auf diese Art und Weise installiert. Idealerweise sollte die Anleitung vorher einmal durchgelesen werden.

Ich übernehme aber selbstverständlich keinerlei Verantwortung/Haftung für irgendwelche Fehler/Probleme, die durch diese Anleitung entstehen könnten. Die Installation erfolgt also eigenverantwortlich durch den jeweiligen Leser.

## 5.1. Sicherung

- Sichern aller Netzwerkeinstellungen, um diese nach der Installation wieder einzurichten, z.B.
    - /etc/hostname
    - /etc/hosts
    - /etc/resolv.conf
    - /etc/network/interfaces
    - …
- falls noch eigene Daten auf dem Server lagern, so sind diese natürlich vorher **zu sichern!** Die folgende Installation installiert den Server komplett neu!
- Debian 9 „Stretch“: Vorberechnung des Netzwerkkartennamens. Achtung: bei einem Fehler bei der Vorberechnung ist das System nach dem Reboot nicht mehr erreichbar und muss über die Rescue-Konsole korrigiert werden)
    - Links:
        - https://major.io/2015/08/21/understanding-systemds-predictable-network-device-names/
    - Berechnung:
        - zunächst per lspci den PCI-Pfad ermitteln, z.B.
            - *# lspci*
                - 01:00.0 Ethernet controller: Intel Corporation 82574L Gigabit Network Connection
        - dann die Architektur des Anschlusses ermitteln (Hotswap-PCI-E-Karten werden mit einem s gekennzeichnet, PCI und normale PCI-E-Karten mit einem p, onboard-Karten in einigen Fällen auch als o)
        - Der Name setzt sich dann wie folgt zusammen:
            - enp0s20 (Online.net Dedibox SC: lspci: 00:14.0)
                - en: Ethernet

        - p: PCI oder normale PCI-E-Karte ohne Hotswap, mit Hotswap: s. In einigen Fällen steht hier ein o für onboard (je nach Anschlußtyp).
        - 0: PCI-Bus 0, der PCI-Bus wird bei lspci durch die ersten 2 Ziffern angezeigt
        - s20: Slot 20, wird bei lspci in HEX in Ziffern 3 und 4 angezeigt.
    - Weitere Beispiele:
        - enp1s0 (Kimsufi KS-1, siehe auch obiges lspci-Beispiel: 01:00.0)

## 5.2. Wechsel in den Rescue-Modus

- Manager-Oberfläche des Hosters aufrufen
    - OVH
        - neu (hier: Sprache im Manager in Englisch):
            - https://www.kimsufi.com/en/manager/
            - ggfs. Server auswählen
            - Netboot klicken
            - Rescue auswählen
            - Next
            - Confirm
            - Restart auswählen und bestätigen
        - alt:
            - Server auswählen
            - Unter „General Information“ Modify auswählen
            - „Boot on rescue mode“ auswählen und bestätigen
            - Restart auswählen und bestätigen
            - Eine eMail mit den Zugangsdaten wird zugestellt
    - Online.net
        - https://console.online.net/en/login
        - Dropdown Server anklicken
        - Server List wählen
        - Server anklicken
        - Rescue anklicken und als OS „Ubuntu 16.04 amd64“ auswählen und per Click starten
        - dann den Server neu starten
        - eine eMail mit den Zugangsdaten sollte kurz darauf erscheinen
        - Hinweise:
            - das gewählte Rescue-OS hatte in meinem Fall eine französische Lokalisierung.
            - Es fehlten benötigte Standard-Tools, die sich aber problemlos wie folgt nachinstallieren liessen:
                - # apt install cryptsetup lvm2 debootstrap
- Anmelden im Rescue-Modus
    - *$ ssh -oUserKnownHostsFile=/dev/null -oStrictHostKeyChecking=no root@ip.ad.res.se*
        - Passwort wechseln mit passwd
- Partitionieren der Festplatte
    - Anmerkung:
        - ich partitioniere wie folgt:
            - sda1: /boot 100MB
            - LVM: /root 8GB

        - LVM: /tmp 1GB
        - LVM: /var 10GB
        - LVM: /opt 4GB
        - LVM swap 512MB
        - LVM: /home Rest
    - *# fdisk /dev/sda*
        - neue Partitionstabelle mit 'o'
        - neue Bootpartition mit 'n'
            - primary mit 'p'
                - erste Partition mit '1'
                - ersten Sektor übernehmen
                - letzter Sektor: erweitern um 100MB mit '+100MB'
            - erste Partition aktivieren mit 'a', dann '1' für erste Partition
        - neue Datenpartition mit 'n'
            - ebenfalls Primary mit 'p'
                - zweite Partition mit '2'
                - ersten und letzten Sektor übernehmen
        - darauf achten, dass beide Partitionen die richtige Media-ID von Hex 83 (Linux) haben (ist eigentlich default. Mit 'p' prüfen)
        - mit 'w' Partitionstabelle schreiben
- Dateisystem für die Boot-Partition anlegen
    - *# mkfs.ext4 /dev/sda1*
- verschlüsselte Partition auf sda2 anlegen
    - *# cryptsetup -s 512 -c aes-xts-plain64 luksFormat /dev/sda2*
        - Achtung: die Cryptoparameter sollten nicht anders gewählt werden, da dieses dieselben sind, die der Debian-Installer verwendet!
        - Auf einem Jessie-System habe ich den Hash-Parameter statt default sha1 auf sha512 gesetzt! → -h sha512
        - Das Passwort für die verschlüsselte Partition sollte mind. 20 Zeichen lang sein und Klein-/Grossbuchstaben, sowie Ziffern enthalten.
    - Ermitteln der UUID der Partition sda2 auf mail0 (wichtig für den späteren Einrichtungsprozess, siehe unten). Die **UUID bitte unbedingt notieren!**
        - *# cryptsetup luksDump /dev/sda2 | grep UUID*
            - Ausgabe:
              *UUID: aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee*
- Öffnen der Crypto-Partition:
    - *# cryptsetup luksOpen /dev/sda2 sda2_crypt*
- Einrichten des Logical Volume Managements (LVM):
    - Erstellen des Physical Volumes:
        - *# pvcreate /dev/mapper/sda2_crypt*
    - Erstellen der Volume Group:
        - *# vgcreate vg0 /dev/mapper/sda2_crypt*
    - Erstellen der logischen Volumes:
        - *# lvcreate -L 8g -n root vg0*
        - *# lvcreate -L 10g -n var vg0*
        - *# lvcreate -L 4g -n opt vg0*
        - *# lvcreate -L 1g -n tmp vg0*
        - *# lvcreate -L 512m -n swap vg0*
        - *# lvcreate -n home -l 100%FREE vg0*

- Dateisysteme mounten, Swapspace einrichten, unter /target mounten. Lesbare Labels verwenden:
  - *# for i in root var opt tmp home; do mkfs.ext4 -L $i /dev/mapper/vg0-$i; done*
  - *# mkswap -L swap /dev/mapper/vg0-swap*
  - *# mkdir /target*
  - *# mount /dev/mapper/vg0-root /target*
  - *# mkdir /target/{boot,home,opt,tmp,var}*
  - *# mount /dev/sda1 /target/boot*
  - *# for i in home opt tmp var; do mount /dev/mapper/vg0-$i /target/$i; done*
  - *# swapon /dev/mapper/vg0-swap*
- richtige Permissions für tmp setzen
  - *# chmod 1777 /target/tmp*
- die Festplatte ist jetzt vorbereitet. Als nächstes ist debootstrap einzusetzen (wird von OVH im Rescue-Mode vorgehalten, bei online.net siehe oben nachzuinstallieren). Bitte jetzt auch den gpg-Keyring überprüfen, um Korrektheit/Unversehrtheit der Debian-Installationspakete zu verifizieren. Hier ein Ausgabebeispiel:
  - *# gpg /usr/share/keyrings/debian-archive-keyring.gpg*
    *pub 4096R/B98321F9 2010-08-07 Squeeze Stable Release Key <debian-release@lists.debian.org>*
    *pub 4096R/473041FA 2010-08-27 Debian Archive Automatic Signing Key (6.0/squeeze) <ftpmaster@debian.org>*
    *pub 4096R/65FFB764 2012-05-08 Wheezy Stable Release Key <debian-release@lists.debian.org>*
    *pub 4096R/46925553 2012-04-27 Debian Archive Automatic Signing Key (7.0/wheezy) <ftpmaster@debian.org>*
- Durchführung der Installation, was einige Minuten dauern kann. Achtung: Angeben der Architektur und des Debian-Spiegels:
  - **Jessie:** *# debootstrap --arch amd64 jessie /target http://ftp.de.debian.org/debian*
  - **Stretch:** *# debootstrap --arch amd64 stretch /target http://ftp.de.debian.org/debian*
- die meisten Dinge sind jetzt getan. Da es sich hier aber um keine Default-Installation handelt, müssen noch einige Dinge manuell angepasst werden, sowie der chroot durchgeführt werden:
  - *# mount -o bind /dev /target/dev*
  - *# mount -t proc proc /target/proc*
  - *# mount -t sysfs sys /target/sys*
  - *# XTERM=xterm-color LANG=C.UTF-8 chroot /target /bin/bash*
- bei Fehlern bezüglich der Locale:
  - *# locale-gen en_US.UTF-8*
  - *# dpkg-reconfigure locales*
- nun muss die crypttab angepasst werden, damit beim Booten das richtige Volume aufgemacht werden kann. Hierfür wird die vorhin (siehe weiter oben) gespeicherte UUID benötigt:
  - *# echo 'sda2_crypt UUID=aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee none luks' > /etc/crypttab*
- damit beim ersten Reboot die Mountpoints verwendet werden können, muss die /etc/fstab entsprechend angelegt werden:
  - *# vi /etc/fstab*

```
# <file system label> <mount point>   <type>   <options>   <dump> <pass>
LABEL=root            /             ext4     errors=remount-ro,relatime  0       1
```

*LABEL=tmp /tmp ext4 rw,nosuid,nodev 0 2*
*LABEL=var /var ext4 rw 0 2*
*LABEL=opt /opt ext4 rw,nodev 0 2*
*LABEL=home /home ext4 rw,nosuid,nodev 0 2*
*/dev/sda1 /boot ext4 rw,nosuid,nodev 0 2*
*LABEL=swap none swap sw 0 0*

- einige Tools benötigen noch /etc/mtab. Hier reicht für gewöhnlich ein Link.
  - *# ln -sf /proc/mounts /etc/mtab*
- erstellen von /etc/network/interfaces. Hier bei Debian Jessie die gesicherte Version verwenden, die anfangs gesichert wurde. Bei Debian Stretch ist hier statt eth0 die hardwarenahe Bezeichnung (siehe Vorberechnung oben) zu verwenden. IPv6 auskommentieren. Diese am Ende der Sektion 'iface eth0 inet static' (bei Stretch wieder die hardwarenahe Bezeichnung) erweitern um (Begründung später):
  - *pre-up /sbin/ip addr flush dev **eth0** || true*
- bei Bedarf die Autokonfiguration von IPv6 abschalten (Achtung: bei Stretch wieder die hardwarenahe Bezeichnung verwenden):
  - *# vi /etc/sysctl.conf*
    *# Disable IPv6 autoconf*
    *net.ipv6.conf.all.autoconf = 0*
    *net.ipv6.conf.default.autoconf = 0*
    *net.ipv6.conf.**eth0**.autoconf = 0*
    *net.ipv6.conf.all.accept_ra = 0*
    *net.ipv6.conf.default.accept_ra = 0*
    *net.ipv6.conf.**eth0**.accept_ra = 0*
- DNS konfigurieren: hier die anfangs gesicherten Daten in die /etc/resolv.conf zurückspielen oder anderweitig anpassen.
- Hostname eintragen
  - *# echo mein.servername.de > /etc/hostname*
- /etc/hosts anpassen (ggfs. Wie hier im Beispiel die IPv6-Einträge auskommentieren):
  - *# vi /etc/hosts*
    *# Do not remove the following line, or various programs*
    *# that require network functionality will fail.*
    *127.0.0.1 localhost.localdomain localhost localhost.meinedomain.de*
    *ip.ad.res.se servername.meinedomain.de*
    *# The following lines are desirable for IPv6 capable hosts*
    *#(added automatically by netbase upgrade)*
    *##::1 ip6-localhost ip6-loopback*
    *##fe00::0 ip6-localnet*
    *##ff00::0 ip6-mcastprefix*
    *##ff02::1 ip6-allnodes*
    *##ff02::2 ip6-allrouters*
    *##ff02::3 ip6-allhosts*
- BIOS-Uhr auf UTC laufen lassen
  - *# echo -e '0.0 0 0.0\n0\nUTC' > /etc/adjtime*
- Zeitzone einstellen:
  - *# dpkg-reconfigure tzdata*
    *Europe/Berlin*
- APT konfigurieren:
  - **Jessie:** *# vi /etc/apt/sources.list*
    *deb http://ftp.de.debian.org/debian jessie main contrib*
    *deb http://ftp.de.debian.org/debian/ jessie-updates main contrib*

*deb http://security.debian.org/ jessie/updates main contrib*
*deb-src http://security.debian.org/ jessie/updates main contrib*

- **Stretch:** *# vi /etc/apt/sources.list*
  *deb http://ftp.de.debian.org/debian stretch main contrib*
  *deb http://ftp.de.debian.org/debian/ stretch-updates main contrib*
  *deb http://security.debian.org/ stretch/updates main contrib*
  *deb-src http://security.debian.org/ stretch/updates main contrib*
- *# apt-get update*

- Fehlende Komponenten, die für den Betrieb bzw. auch den Startprozess des Servers nötig sind, nachinstallieren:
  - *# apt-get install makedev cryptsetup lvm2 ssh dropbear busybox ssh initramfs-tools locales linux-image-amd64 grub-pc kbd console-setup screen*
- Es wird grub installiert (sollte nach /dev/sda gehen). ConsoleFont habe ich auf „guess“ gelassen.
- Vor dem Installieren von Paketen sicher stellen, dass die initiale RAM-Disk (initrd) uns erlaubt, uns zu konnektieren. Ein Root-Passwort wird nicht möglich sein. Daher sollte eine vorbereitete authorized_keys-Datei mit dem eigenen Public-Key übertragen werden.
  - *# scp -oUserKnownHostsFile=/dev/null -oStrictHostKeyChecking=no /Quelle/zur/eigenen/authorized_keys-Datei servername.meinedomain.de:/target/etc/initramfs-tools/root/.ssh/*
- **Stretch**: seit Stretch muss die authorized_keys auch noch nach /etc/dropbear-initramfs/ kopiert werden, sonst kommt es zu Fehlern bei update-initramfs -uv. In folgender Art:
  - *dropbear: WARNING: Invalid authorized_keys file, remote unlocking of cryptroot via SSH won't work!*
- wenn sich obiger Schlüssel oder aber etc/dropbear/dropbear_*_host_key, der Inhalt von /etc/crypttab oder eine andere Schlüsselkomponente für den Bootprozess sich ändert, so ist folgendes aufzurufen, um die initrd wieder auf den aktuellen Stand zu bringen:
  - *# update-initramfs -uv*
- Nachdem grub installiert wurde, ist auch die initrd erstellt worden. Hier sollte deren Inhalt, z.B. keys geprüft werden:
  - **Jessie:** *# zcat /boot/initrd.img-3.16.0-4-amd64 | cpio -t conf/conf.d/cryptroot etc/lvm/lvm.conf etc/dropbear/* root/.ssh/authorized_keys sbin/dropbear*
  - **Stretch:** # zcat /boot/initrd.img-4.* | cpio -t conf/conf.d/cryptroot etc/lvm/lvm.conf etc/dropbear/* root/.ssh/authorized_keys sbin/dropbear
  - Ausgabe:
    *root/.ssh/authorized_keys*
    *etc/dropbear/dropbear_dss_host_key*
    *etc/dropbear/dropbear_rsa_host_key*
    *etc/lvm/lvm.conf*
  - ACHTUNG: hier fehlt unter Umständen conf/conf.d/cryptroot !!!
    - gegensteuern (hier wieder die gespeicherte UUID einsetzen):
      - *# vi /etc/initramfs-tools/conf.d/cryptroot*
        *target=sda2_crypt,source=UUID=aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee,key=none,rootdev,lvm=vg0-root*
      - *# update-initramfs -uv*
      - kontrollieren mit zcat (siehe oben)
- Der Cryptroot muss hier den richtigen Inhalt haben. Daher den Inhalt noch einmal kontrollieren:
  - *# zcat /boot/initrd.img-* | cpio -i --to-stdout conf/conf.d/cryptroot*
    - Ausgabe:

*target=sda2_crypt,source=UUID=aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee,key=none,rootdev,lvm=vg0-root*

- jetzt muss dem Kernel noch mitgeteilt werden, so früh wie möglich das Netzwerk bereitzustellen, um die Partitionen aufzuschließen. Dies geschieht schon in der grub.cfg
  - *# vi /etc/default/grub*
    - *GRUB_CMDLINE_LINUX="ip=ip.ad.res.se::default.gateway.ad.resse:netz.mas.ke::eth0:none"*
    - Beispiel:
      - *GRUB_CMDLINE_LINUX="ip=1.1.1.1.2::1.1.1.1:255.255.255.0::eth0:none"*
  - Syntax hier: linux ip=<ipaddress>:[<dnsserver>]:<gateway>:<netmask>:<hostname>:<network-interface>:{off|on|dhcp6|auto6|none}
  - abschalten der quiet-Option für den Fall, dass doch einmal der Boot überwacht werden muss. Es wird in einigen Quellen eine ominöse QEMU-Bootvariante erwähnt, die ich aber noch nicht probiert habe.
    - *GRUB_CMDLINE_LINUX_DEFAULT=""*
- Im Fall des Hosters online.net:
  - Link:
    - https://documentation.online.net/en/dedicated-server/hardware/configure-ipmi-server/serial-console-suchard
  - damit die „serielle Konsole“ der Web-GUI genutzt werden kann, sollten noch folgende Anpassungen in /etc/default/grub durchgeführt werden:
    *GRUB_TERMINAL=serial*
    *GRUB_SERIAL_COMMAND="serial --unit=1 --speed=9600 --word=8 --parity=no –stop=1"*
  - Zusätzlich ist folgendes nötig:
    - *# systemctl enable getty@ttyS1.service*
    - *# systemctl start getty@ttyS1.service*
- Aktivieren der neuen grub-Konfiguration:
  - *# update-grub2*
- Automatisches Beheben von Dateisystemproblemen kann ebenfalls helfen:
  - *# echo FSCKFIX=yes >> /etc/default/rcS*
- nützliche weitere Pakete installieren:
  - *# apt-get install vim less ntpdate sudo*
- User hinzufügen und anpassen:
  - *# useradd -u 500 -g 100 -d /home/deinUsername -s /bin/bash -c "Dein Kommentar" -m deinUsername*
  - *# passwd deinUsername*
  - *# mkdir /home/deinUsername/.ssh*
  - *# chmod 700 /home/deinUsername/.ssh*
  - *# chown deinUsername:users /home/deinUsername/.ssh*
  - *# cp /etc/initramfs-tools/root/.ssh/authorized_keys /home/deinUsername/.ssh/*
    - (hier ggfs. auch einen anderen Public-Key einsetzen)
  - *# chown deinUsername:users /home/deinUsername/.ssh/authorized_keys*
  - *# chmod 600 /home/deinUsername/.ssh/authorized_keys*
- Rootpasswort setzen
  - *# passwd*
- Um den SSH-Server etwas unanfälliger für Angriffe zu machen, sollte die sshd_config angepasst werden (einige Optionen sind zu ändern, andere hinzuzufügen):

- *# vi /etc/ssh/sshd_config*
  - *Port 2222*
    - damit Angriffe auf den Standardport 22 ins Leere laufen
  - *PasswordAuthentication no*
    - Zugriff NUR mit dem Public-/Private-Key-Verfahren
  - *Debianbanner no*
    - damit wird der SSH-Server daran gehindert, sich als Debian-Variante zu outen
  - Kommentar entfernen von: *ListenAddress 0.0.0.0*
- dropbear-sshd aus Standard-Bootsequenz entfernen, damit nicht gleichzeitig zwei SSHd-Instanzen aktiv sind. Den Dropbear benötigen wir nur für die initrd.
  - *# update-rc.d -f dropbear remove*
  - prüfen:
    - *# ls -la /etc/rc2.d/*
    - hier sollte der dropbear nicht mehr mit einem S am Anfang zu sehen sein.
  - ist eigentlich nicht nötig, da Debian das erkennt und Dropbear nicht startet (siehe /etc/default/dropbear)
- Dropbear in /etc/initramfs-tools/initramfs.conf anpassen
  - *DROPBEAR=y*
  - (siehe auch der Hook in /usr/share/initramfs-tools/hooks/dropbear)
  - **Stretch:** DROPBEAR=y ist deprecated und sollte so in Stretch NICHT mehr gesetzt werden!
- Dropbear-Port ändern:
  - *# vi /etc/default/dropbear*
    *DROPBEAR_PORT=2222*
  - *# vi /etc/dropbear/run*
    *exec dropbear -d ./dropbear_dss_host_key -r ./dropbear_rsa_host_key -F -E -p 2222*
  - *# cp /usr/share/initramfs-tools/scripts/init-premount/dropbear /etc/initramfs-tools/scripts/init-premount/*
  - *# vi /etc/initramfs-tools/scripts/init-premount/dropbear*
    - am Ende der Conf-Datei für den Port anpassen:
      */sbin/dropbear -s -p 2222*
    - **Stretch:**
      - /etc/initramfs-tools/scripts/init-premount/dropbear ist nicht mehr anzupassen! Der Port kommt jetzt aus den $DROPBEAR_OPTIONS
      - stattdessen:
        - *# cd /etc/dropbear-initramfs/*
        - *# sed -e "s/^\(#\)\?\(DROPBEAR_OPTIONS=\).*$/\2'-p 2022'/g" -e 's|^\(#\)\?\(\(IFDOWN=\).*\)$|#\2\n\3none|' -i /etc/dropbear-initramfs/config*
  - wieder die initrd aktualisieren:
    - *# update-initramfs -uv*

## 5.3. Verlassen des Rescue-Modus und Starten des Servers

- Es ist fast geschafft. Der Server ist nun vorbereitet.
- Beenden der chroot-Umgebung:
  - *# exit*
  - *# umount /target/{dev,proc,sys,boot,home,tmp,opt,var}*
  - falls das Unmounten nicht klappt (speziell dev und proc), so hat sich vermutlich der irqbalance-Daemon bei der Paketinstallation eingeschmuggelt. Diesen dann einfach via

SIGTERM abschiessen und dann dev und proc noch einmal unmounten. Abschiessen z.B. via:
        - *# kill -SIGTERM 28443* (wobei 28443 hier die PID des irqbalance-Daemons darstellt)
    - *# umount /target*
    - *# swapoff -a*
    - *# lvchange -an /dev/mapper/vg0-**
    - *# cryptsetup luksClose sda2_crypt*
- Manager-Oberfläche von OVH aufrufen
    - neu (hier: Sprache im Manager in Englisch):
        - https://www.kimsufi.com/en/manager/
        - ggfs. Server auswählen
        - Netboot klicken
        - Hard disk auswählen
        - Next
        - Confirm
        - Restart auswählen und bestätigen
    - alt:
        - Server auswählen
        - Unter „General Information“ Modify auswählen
        - „Boot on the hard disc“ auswählen und bestätigen
- Restart auswählen und bestätigen
- Ein Ping auf den Server sollte nach einiger Zeit antworten. Zunächst kam bei mir eine Mail, dass die Remote-Hardware-reboot-Anfrage fehlerhaft war und ein Techniker das in den nächsten Minuten beheben würde.
    - 400 Pings gingen verloren, dann war der Server wieder pingbar. Ein Techniker hat den Server offenbar hart resettet. Das ist jedoch nur einmal bei mir vorgekommen.

## 5.4. Erstanmeldung am neuen Server

- Hinweise zur ersten Anmeldung am Dropbear:
    - via ssh auf Port 2222 verbinden und sogleich die eigene lokale Datei ~/.ssh/known_hosts auf dem eigenen privaten PC wegsichern, da der Dropbear-Hostkey anders ist, als der des OpenSSH-Servers, der nach dem Aufschließen des Cryptocontainers verwendet wird:
        - Achtung: nicht mit User deinUsername konnektieren, da dieser im Dropbear nicht bekannt ist (ein -vvv liefert hier: PEM_read_PrivateKey failed). Hier den root-User verwenden. Dieser hat weiter oben ja auch den SSH-public-Key bekommen.
    - Auf dem eigenen PC: *$ cp .ssh/known_hosts .ssh/known_hosts.eigenerServername*
        - in der gerade gesicherten Datei known_hosts.initramfs_eigenerServername alle Einträge, bis auf den eben neu vom Dropbear erhaltenen Hostkey, löschen

## 5.5. Entsperren des Cryptocontainers beim Booten

Nach dem Bootvorgang wird erst einmal eine Minimalumgebung aus der initialen RAM-Disk (initrd) gestartet, in der der Dropbear aktiv ist. Nach Eingabe der Passphrase für den Cryptocontainer wird selbiger aufgeschlossen und der Bootvorgang wird automatisch fortgesetzt. Nach erfolgtem Boot kann man sich dann normal am Server anmelden.

- Um den Cryptocontainer aufzuschließen bieten sich 3 Wege an:
  - Automatisiertes Aufschließen per Skript:
    - *# vi unlock_server.sh*
      - *ssh -p 2222 -o "UserKnownHostsFile=~/.ssh/known_hosts.eigenerServername" -i "~/.ssh/id_dsa" root@eigenerServer "echo -ne \"geheime_Passphrase\" >/lib/cryptsetup /passfifo"*
  - Manuelles Aufschließen mit Passphraseangabe (Achtung: hier wird die Passphrase im Klartext angezeigt):
    - *$ ssh -p 2222 -o UserKnownHostsFile=~/.ssh/known_hosts.eigenerServername root@eigenerServer 'cat > /lib/cryptsetup/passfifo'*
      - Achtung: hier kein Enter eintippen nach dem Passphrase, sondern zweimal CTRL-D !!!
  - Manuelles Entsperren des Cryptocontainers auf dem Server in der Busybox
    - # echo -n “geheimes Passwort“ > /lib/cryptsetup/passfifo
    - der Server setzt dann seinen Bootvorgang fort.
- Workaround zum Aufschließen des Cryptocontainers bei einer **frischen Jessie-Installation**:
  - Da (Stand 05/2015) bei einer frischen Jessie-Installation das passfifo-Tool fehlt (unter /lib/ kein cryptsetup-Verzeichnis!), muss man sich manuell in der Busybox anmelden (obiger 3. Punkt). Danach sind folgende Schritte nötig:
    - manuelles Entsperren des Cryptocontainers:
      - *# /sbin/cryptsetup luksOpen /dev/sda2 sda2_crypt*
    - jetzt hängt jedoch der weitere Boot in den Cryptocontainer hinein. Es ist das Cryptroot-Skript manuell zu beenden:
      - *# ps*
      - *# kill -9 <Prozess-ID des cryptroot-Prozesses>*
      - Beispielausgabe:

        *~ # /sbin/cryptsetup luksOpen /dev/sda2 sda2_crypt*
        *Enter passphrase for /dev/sda2:*

        *~ # ps*
        *...*
        *131 root {cryptroot} /bin/sh /scripts/local-top/cryptroot*
        *136 root /lib/cryptsetup/askpass Please unlock disk sda2_crypt:*
        *137 root /sbin/cryptsetup -T 1 open --type luks /dev/disk/by-uuid/.....*
        *...*
        *~ # kill -9 131*
    - jetzt bootet auch Jessie in den Cryptocontainer!
    - Nach dem Starten in den Cryptocontainer habe ich bei mir noch die übriggebliebenen obigen 2 Prozesse gefunden, die ich per SIGKILL beendet habe:
      - *# ps ax*

        *...*
        *136 ? S 0:00 /lib/cryptsetup/askpass Please unlock disk sda2_crypt:*
        *137 ? S<L 0:00 /sbin/cryptsetup -T 1 open --type luks /dev/disk/by-uuid/…*
        *…*
      - *# kill -SIGKILL 136 137*

# 6. Verbesserungsideen

- 2-Faktor-Authentisierung
  - Yubikey-Support
  - Zertifikatsbasierte Authentisierung