Dokumentation zu Quiddi-github

Quiddi

Inhaltsverzeichnis

1 Git 5
1.0.1 neues Projekt 5
1.0.2 GitHub Upload 5
1.1 eigener Git Hub Server 6

2 Verschlüsseln von Passwörtern 7
2.1 GPG Schlüssel importieren exportieren erstellen 7
2.1.1 Export Keys 7
2.1.2 Import Keys 7
2.1.3 neues Key-Paar erstellen 8
2.2 pass 8

3 Microsoft Office 2013 9
3.1 Wine installieren 9
3.2 andere Pakete 9
3.3 Konfiguration Wine 10

4 DarkNet 11

5 VMware 13
5.0.1 Linux als Guest System 13

6 Debian 15

7 Bluetoth Lautsprecher 17

8 Dynamische IP updaten 19

9 Feste IP einstellen mit dhcpcd 21

10 SSH mit RSA-Keyfile 23

10.1 Keyfile am Client einspielen 23

1. Git

1.0.1 neues Projekt

In den Ordner im Terminal reinnavigieren und mit

```
git init
```

das Git-Projekt initialisieren. Danach:

```
git add *.*
git add README
git commit –m 'Projekt neu angelegt'
```

1.0.2 GitHub Upload

Danach kann auf der GitHub Homepage ein neues Projekt angelegt werden. Um dies pushen zu können muss man

```
git remote add origin https://github.com/quiddi-github/linux.bibel.git
```

origin adden. Danach kann man das Projekt auf origin pushen, mit:

```
git push –u origin master
```

Heirmit wird es auf dem Master-Branch gepushed. Zuvor müssen aber noch alle Dateien reingeladen werden mit:

```
git add .
```

und ein Kommentar angelegt werden mit

```
git commit –m 'Projekt neu angelegt'
```

Die remote Reposities können mit

```
git remote show
```

bzw. wenn ein remote Reposity namens origin vorliegt, dies mit

```
git remote show origin
```

anzeigen lassen.

1.1 eigener Git Hub Server

Git sollte installiert sein. Der Rest läuft dann wie lokal. Am Server sollte nun folgendes passieren:
1. ordner in /home/git/usb_drive als xxx.git erstellen
2. git –bare init ausführen im ordner
Am Client nun folgendes machen:

- Neuen Ordner anlegen $ cd myproject
- Initialisieren $ git init
- Dateien hinzufügen mit $ git add .
- Kommentar zu den Änderungen schreiben $ git commit -m 'initial commit'
- Server nur einmal konfigurieren $ git remote add origin git@gitserver:/opt/git/project.git
- Git push ausführen $ git push origin master

2. Verschlüsseln von Passwörtern

2.1 GPG Schlüssel importieren exportieren erstellen

GPG verwaltet unter Linux die Schlüssel.
Man kann sich die **öffentlichen** Schlüssel anzeigen lassen mit:

```
gpg --list-keys
```

oder die **privaten** mit:

```
gpg --list-private-keys
```

2.1.1 Export Keys

Zum exportieren bedient man sich folgendem Befehl:
öffentlich

```
gpg -a --output public.gpg --export <Key-ID>
```

für den **privaten**:

```
gpg --export-secret-keys -a <Key-ID> > secret.asc
```

2.1.2 Import Keys

Die Schlüssel müssen nun auf dem anderen PC importiert werden, Dies passiert für den **öffentlichen** Schlüsssel mit

```
gpg --import public.gpg
```

und für den **privaten** Schlüssel mit

```
gpg --allow-secret-key-import --import secret.asc
```

Um mit dem öffentlischen Schlüssel abreiten zu können, muss diesem noch vertraut werdem. Andernfalls erhält man z.B. in dem Programm pass eine Fehlermeldung. Zum vertrauen des Schlüssels geht man wie folgt vor:

```
gpg --edit-key <Key-ID>
```

Danach öffent sich ein seperates Programm im Terminal. Hier gibt man „trust“ ein und bestätigt dies mit „5 “ für „Ich vertraue ihm absolut“. Danach kann man es wieder schließen. Danach kann mit dem öffentlichen Schlüssel gearbeitet werden. Der private Schlüssel wird nur zum entschlüsseln gebraucht und ihm muss daher nicht vertraut werden.

2.1.3 neues Key-Paar erstellen

Ein neues GPG-Key-Paar, also der öffentliche- mit einem provatem Schlüssel, wird erzeugt mit

```
gpg --full-generate-key
```

Danach muss „1“ gedrückt werden um einen RSA-Key zu erhalten. Danach wird nach der bit-Größe gefragt. Hier kann z.B. „1024“ eingegeben werden. Danach wird nach dem Ablaufdatum gefragt. Ist das Ablaufdatum erreicht so kann mit dem öffentlichen Schlüssel nichts neues mehr verschlüsselt werden. Es können aber noch 100 Jahre nach dem Ablaufdatum vorhandene verschlüsselte Dateien entschlüsselt werden. Er frägt dann weiter ob die Daten korrekkt sind, Nach einem Namen, e-Mail, Kommentar. Danach wird nach einem Passphrase gefragt. Der Passphrase ist notwendig um den Key nachher benutzen zu können. Ohne den Passphrase zu wissen kann er später nicht benutzt werden. Danach ist der Key erstellt.

2.2 pass

Das Programm pass verschlüsselt, mit dem zuvor defineirten öffentlichen Schlüssel, Text, und entschlüsselt dies mit dem dazugehörigen private Key wieder. Hat man ein Schlüsselpaar, bestehend aus öffentlichem- und privatem Schlüssel erstellt so kann man dieses mit

```
pass init <Key-ID>
```

für pass defineren. Achtung, wurde der Key zuvor importiert, so muss dem öffentlichen Key erst vertraut werden, ansonsten kann ein verschlüsseln mit pass nicht stattfinden. Sollte man nun alte verschlüsselte Dateien haben, so können diese nun in „*/home/USER/.password-store*“ kopiert werden. Das entschlüsseln sollte nun mit

```
pass <verschluesselte Datei>
```

funktionieren. Mit „pass“ werden alle verschlüsselten Inhalte angezeigt. Mit „pass insert <verschlüsselte Datei>“ kann dies dann wieder im Klartext dargestellt werden. Mit „pass rm <verschlüsselte Datei>“ kann das Teil wieder gelöscht werden. Mehrere Zeilen können mit

```
pass insert -m <verschluesselte Datei>
```

angelegt werden.

3. Microsoft Office 2013

Vorraussetzungen:

- Office 2013 als 32-bit Version
- Die Pakete: wine winetricks mono-devel samba-winbind

3.1 Wine installieren

Wine wird in Debian wie folgt installiert:
32-Bit Architektur hinzufügen:

```
sudo dpkg --add-architecture i386
```

Den entsprechenden Key hinzufügen

```
wget -nc https://dl.winehq.org/wine-builds/Release.key
sudo apt-key add Release.key
```

Danach kann in die */etc/apt/sources.list* folgendes hinzugefügt werden:

```
deb https://dl.winehq.org/wine-builds/debian/ DISTRO main
```

DISTRO ist hierbei urch das Debian System zu erssetzen. Danach das ganze updaten:

```
sudo apt-get update
```

Nun die StableVersion von Wine installieren:

```
sudo apt-get install --install-recommends winehq-stable
```

3.2 andere Pakete

Es fehlern nun noch
winetricks mono-devel samba-winbind
Diese bitte noch installieren. Am einfachsten mit

```
sudo apt-get install winetricks mono-devel winbind
```

Das war soweit die Vorbereitung. Nun kann Wine konfiguriert werden

3.3 Konfiguration Wine

Für Office 2013 einen neuen Ordner anlegen:

```
WINEPREFIX=~/.wine/Office2013 WINEARCH=win32 winecfg
```

Es muss Windows 7 verwendet werden. Nun noch folgendes nachinstallieren

```
WINEPREFIX=~/.wine/Office2013 WINEARCH=win32 winetricks corefonts
WINEPREFIX=~/.wine/Office2013 WINEARCH=win32 winetricks msxml4
WINEPREFIX=~/.wine/Office2013 WINEARCH=win32 winetricks msxml6
WINEPREFIX=~/.wine/Office2013 WINEARCH=win32 winetricks riched30
WINEPREFIX=~/.wine/Office2013 WINEARCH=win32 winetricks vb6run
WINEPREFIX=~/.wine/Office2013 WINEARCH=win32 winetricks d3dx11_42
WINEPREFIX=~/.wine/Office2013 WINEARCH=win32 winetricks d3dx11_43
```

Es soll nun in der Konfig bei Bibliotheken unter Bestehende ein Eintrag „*riched20 (Native, Builtin)“ sein. Die kOnfig kann mit

```
WINEPREFIX=~/.wine/Office2013 WINEARCH=win32 winecfg
```

aufgerufen werden. Nun kann Office 2013 installiert werden

```
WINEPREFIX=~/.wine/Office2013 WINEARCH=win32 wine /PATH_TO_MOUNTPOINT/setup.e
```

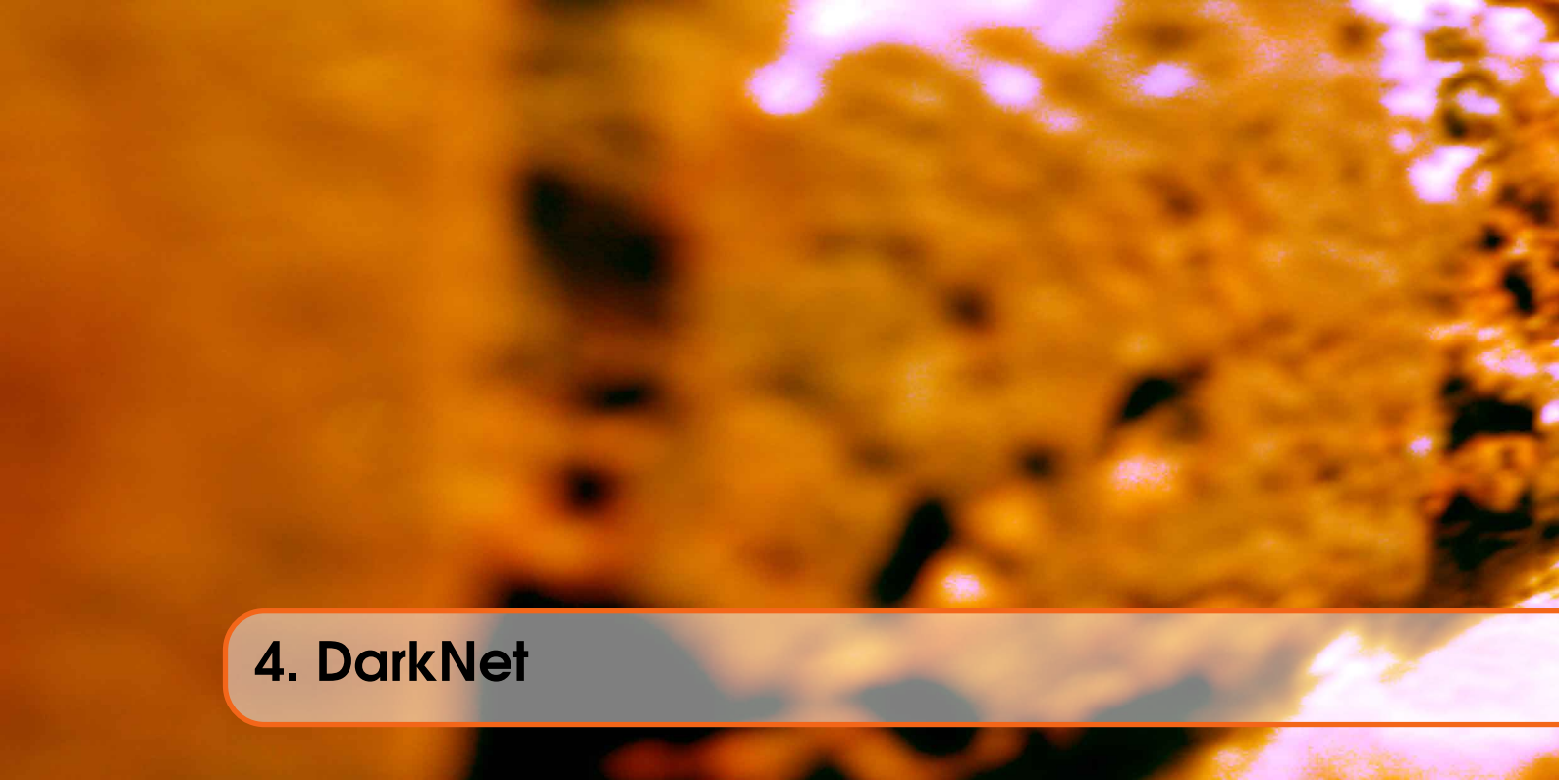
Bei mir bleib die Installation irgendwann hängen, starten tut mein Word,... trotzdem. Word kann nun gestartet werden mit

```
WINEPREFIX=~/.wine/Office2013 WINEARCH=win32 wine "C:\\Program Files\\Microso
```

Hierbei wird es wahrscheinlich zu einem schwarzen Bildschirm kommen. Dies kann man folgendermaßen gerichtet werden. Wir machen zunächst die Regedit auf

```
WINEPREFIX=~/.wine/Office2013 WINEARCH=win32 wine regedit
```

Unter „HKEY_CURRENT_USER{}Software{}Wine“ legen wir den neuen Schlüssel „Direct3D“ an. Danach wird ein neuer DWORD Wert mit dem Namen „MaxVersionGL“ angelegt. Als Hexwert schreiben wir hier „30002“ hinein. Sollte dies immer noch nicht funktionieren soo wird noch unter „HKEY_CURRENT_USER{}Software{}Wine“ ein Schlüssel mit „Direct2D“ angelegt. Auch hier muss wieder ein DWORD angelegt werden mit dem Namen „max_version_factory“. Hier sollte dann der Wert „0“ drinnenstehen. Das war’s



4. DarkNet

Auf die Funktionen, wie das DarkNet funktionert, soll hier nicht eingegangen werden. Ea kann unter Linux Tor installiert werden. Wird dieser gestartet, so wird lokal(127.0.0.1) ein Proxy auf Port 9050 gestartet. Sofern nicht anderst konfiguriert startet er mit dem SOCKS Protokoll. Zum starten einfach tor eingeben im Terminal. Vereinzelt kann es vorkommen, dass die Anwedung nicht richtig startet. Mit

```
sudo netstat -plnt | fgrep 9050
```

kann geschaut werden was Port 9050 blockiert. Andernfalls kann auch mit

```
sudo killall tor
```

der Tor-Service beendet werden. Chrome kann nunb bei einstellen des richtigen Proxys sofort .onion Domains darstellen. Im Firefox muss der Proxy Socks remote dns einmal aktiviert werden. Hierzu

```
about:config
```

die Konfiguration aufmachen und „network.proxy.socks_remote_dns“ auf true setzen. Danach muss auch noch aktiviert werden, dass .onion Domains aufgelöst werden, dazu den Wert „network.dns.blockDotOnion“ auf false setzen. Mit dem Link

```
https://check.torproject.org/
```

kann geschaut werden, ob man richtig mit dem Tor-Netzwerk verbunden ist. Viel Spaß bei erkunden des DarkNets.

5. VMware

5.0.1 Linux als Guest System

Mount Share Folder

In VMware Workstation muss für die Linux-VM unter „Settings/Options/Shared Folders“ der geteilte Ordner mit dem HostSystem angelegt und aktiviert sein. Zur Linux Guest VM: Es müssen die Programme

```
open-vm-tools
open-vm-tools-desktop
open-vm-tools-dkms
```

installiert sein. Auf einem DebianBasiertem System geht dies mit:

```
sudo apt-get install open-vm-tools open-vm-tools-desktop
                     open-vm-tools-dkms
```

Danach kann der Ordner in der VM gemountet werden. Dies geschieht mit:

```
vmhgfs-fuse .host:dokumente /home/quiddi/shares/
```

Wobei hier „dokumente“ für den freigegebenen Ordnernamen steht und „/home/quiddi/shares/“ für das Verzeichnis auf dem GuestSystem, in welches gemountet wird.

6. Debian

Debian Version anzeigen:

```
cat /etc/issue
```

```
cat /etc/debian_version
```

7. Bluetoth Lautsprecher

Um einen Bluetoth Lautsprecher zu verbinden, werden die Pakete pulseaudio-alsa, pulseaudio-bluetooth, bluez, bluez-libs, bluez-utils benötigt. Diese bitte installieren.

```
pulseaudio-alsa pulseaudio-bluetooth bluez bluez-libs bluez-utils
```

Nach der Installation mit systemctl den entsprechenden Dienst mit

```
sudo systemctl start bluetooth.service
```

starten. Im Terminal den Lautsprecher verbinden. Hierzu

```
bluetoothctl
```

starten. Danach nach Geräten suchen lassen mit

```
[bluetooth]# power on
[bluetooth]# agent on
[bluetooth]# default-agent
[bluetooth]# scan on
```

Irgendwo sollte das Bluetooth Gerät nun auftauchen. Im besten Falle sieht man nun auch einen Namen des Gerätes. Nun das Gerät mit der Geräte-ID verbinden

```
[bluetooth]# pair 00:1D:43:6D:03:26
```

Nun nach dem Pairen verbinden

```
[bluetooth]# connect 00:1D:43:6D:03:26
```

Die ganze Sache beenden mit

```
[bluetooth]# scan off
[bluetooth]# exit
```

Um die Lautstärke nun vom Terminal aus steuern zu können, muss man noch

```
pulsemixer
```

installieren. Danach dies mit

```
pulsemixer
```

starten. Das Gerät ausswählen und Musik hören.
Bei Problemen kann man noch

```
pulseaudio-alsa
```

installieren. Dies sollte möglich machen, dass man mit Alsamixer das PulseaudioGerät steuern kann. Dies klappte bei mir jedoch nicht.

8. Dynamische IP updaten

Kostenlos wird dies momentan nur noch von `ddnss.de` bzw. `ddnss.ch` angeboten. DynDNS kostet mittlerweile Geld. Vorraussetzung is, dass ein ddnss-Benutzer eingerichtet ist. Weiter muss ddclient installiert sein. Nun kann es mit

```
sudo dpkg-reconfigure ddclient
```

eingerichtet werden. Hier wird nun eingegeben:

- anderen Anbieter auswählen
- Name des Servers: ddnss.de
- Protokoll dyndns2
- Benutzer: testbenutzer
- PW eingeben
- Netzwerkschnittstelle, normal eth0
- Vollständige Adresse: meineadresse.ddnss.ch
- PPP-Verbindung, Nein
- als Daemon starten, JA
- Intervall, 300

Der status heirzu kann mit

```
ddclient -query
```

abgefragt werden. Nun sollte noch umgestellt werden, dass ddnss die externe IP und nicht die interne IP bekommt. Hierzu mit dem Editor nano die config verändern mit

```
sudo nano /etc/default/ddclient
```

Hier muss dann die „use=if“auf

```
protocol=dyndns2
use=web, web=my.ip.fi/
server=ddnss.de
```

umgestellt werden. Danach sollte der Dienst ddclient die externe IP an ddnss.de weiterschicken. Um regelmäßig die IP Adresse an ddnss zu schicken muss man in der Datei „/etc/default/ddclient“

```
run_daemon="false" auf true setzen
run_dhclient="true" auf false setzen
run_ipup="true" auf false setzen
```

um alle 300 Sekunden die IP-Adresse an ddnss zu schicken. Einmalig kann man den Dienst mit

```
sudo ddclient -debug -verbose -noquiet
```

starten.

9. Feste IP einstellen mit dhcpcd

```
sudo nano /etc/dhcpcd.conf
```

Hier sollte sehen als bsp.:

```
interface eth0
static ip_address=192.168.0.4/24
static routers=192.168.0.1
static domain_name_servers=192.168.0.1
```

10. SSH mit RSA-Keyfile

10.1 Keyfile am Client einspielen

```
$ mkdir ~/.ssh
$ cat ~/id_rsa.pub >> ~/.ssh/authorized_keys
$ rm ~/id_rsa.pub
$ chmod 600 ~/.ssh/authorized_keys
```

Dann eine Kongig-Datei für SSH anlegen

```
nano /home/quiddi/.ssh/config
```

Hier dann reinschreiben:

```
Host testhost
  HostName testhost.ddnss.ch
  Port 1234
  User ssh-allowed-UserName
  IdentityFile /home/quiddi/.ssh/authorized_keys
  IdentitiesOnly yes
```

nun kann mit
ssh testhost
eine ssh verbindung aufgebaut serden