Dokumentation zu Quiddi-github

Quiddi

Inhaltsverzeichnis

1. Git

1.0.1 neues Projekt

In den Ordner im Terminal reinnavigieren und mit

```
git init
```

das Git-Projekt initialisieren. Danach:

```
git add *.c
git add README
git commit -m 'initial project version'
```

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.