# Sicher umziehen

## Was zu tun ist, wenn der alte PGP-Schlüssel nicht mehr gefällt

> Manchmal muss einfach ein neuer PGP-Schlüssel her – vielleicht ist der alte zu kurz, vielleicht ist er zu alt. Oder vielleicht ist der Polizeispitzel mit Schlüssel und Passphrase („Mantra") durchgebrannt. Klar ist ein neuer Schlüssel schnell gemacht, aber woher sollen Menschen, die euch oder euerer Gruppe Mails schicken wollen, dann wissen, welchen Schlüssel sie verwenden sollen? Blöd wäre ja z.B., wenn der Polizeispitzel den neuen Schlüssel gemacht hätte und dann er die Mails liest und nicht ihr. Es gibt ein paar Handgriffe, die Unfällen beim Schlüsselmanagement vorbeugen. Um die geht es in diesem Artikel.

Die Hintergründe von Unterschriften unter Schlüsseln und dem Web of Trust haben wir in RHZ 3/09 (Verschlüsseln mit Stil[1]) und RHZ 4/09 (Vertrauen unter GenossInnen[2]) diskutiert. Damals endeten wir mit der Überlegung, allzu großer Aufwand beim Schlüsselmanagement sei vielleicht übertrieben, solange die Staatgewalt noch hilfloser mit der Technik kämpft als viele von uns, mensch solle sich aber doch allmählich mit dem Web of Trust vertraut machen, „denn wenn die Gegenseite anfängt, unsere Verschlüsselung anzugreifen, schadet es bestimmt nicht, wenn wenigstens wir in der Roten Hilfe einen kühlen Kopf bewahren."

Das war vor den Snowden-Leaks, bei denen PGP eine große Rolle spielte, groß genug, dass ernsthafte Angriffe des Staates – Unterschieben falscher Schlüssel, Kompromittierung von Schlüsseln und Ähnliches – in den Bereich des Vorstellbaren geraten. Auch deshalb diskutieren wir hier den nächsten Schritt beim Schlüsselmanagement.

Vorweg eine Kurzfassung dessen, was in den alten Artikeln steht: Mensch kann mit PGP auch Dinge unterschreiben. So eine Unterschrift ist der Beleg, dass der_die Unterschreibende den verwendeten Schlüssel „hat". Nur, weil auf dem Schlüssel „Rote Hilfe Oberammergau" steht, heißt das allerdings noch lange nicht, dass sich diese Person dann auch wirklich der Solidarität gegen politische Repression im Voralpenland widmet. Draufschreiben kann das jede_r.

So eine „Identität" zu bestätigen geht eigentlich wie im echten Leben: Wenn ein_e vertrauenswürdige_r Genoss_in sagt, dass wer von der RH OA ist, dann werdet ihr das glauben. Im PGP-Universum ist dieses „sagen, das wer eine Identität hat" wiederum eine Unterschrift, und zwar *unter den Schlüssel*. Wenn ihr dem_der Unterschreibenden vertraut, könnt ihr auch der Identität vertrauen.

Im Bereich der Roten Hilfe unterschreibt u.a. die Datenschutzgruppe Schlüssel von BuVos und OGen – demnächst auf der BDV ist wieder Gelegenheit dazu. Wenn ihr darauf vertraut, dass wir ordentlich nachprüfen, ob Schlüssel, die wir unterschreiben, tatsächlich zu den Gruppen oder Funktionen gehören, könnt ihr anfangen, den entsprechenden Schlüsseln zu vertrauen, indem ihr mit eurer Schlüsselverwaltung nachseht, ob eine Unterschrift von uns auf dem Schlüssel ist.

Aber woher wisst ihr, ob ihr unseren Schlüssel habt oder nicht das, was auf eurer Platte ist, vom VS in unserem Namen ausgestellt wurde, damit ihr Schlüsseln vertraut, die die Schlapphüte mitlesen können? Nun, dafür drucken wir seit über 10 Jahren unter jeden unserer Artikel den Fingerabdruck unseres Schlüssels. Wenn also nicht der VS in einer einzigartigen Kommandoaktion euer RHZ-Archiv ausgetauscht hat, könnt ihr ziemlich zuverlässig prüfen, ob unser Schlüssel auch der ist von den Nerds, die über all die Jahre in der RHZ genervt haben.

Und da kommt jetzt das Problem: Wir müssen unseren Schlüssel ändern. Unser Hauptschlüssel ist nämlich ein 1024D-Schlüssel. Was für normale Menschen toll technisch und bestimmt sicher wirken mag, wird Kryptograf_innen aufjaulen lassen: „Was, *darauf* baut ihr euere Identitässicherung auf??? Nächstes Jahr knackt sowas die NSA wie ne Haselnuss!"

„Knacken" heißt in diesem Zusammenhang, dass andere unsere Unterschrift fälschen können. Wir brauchen also einen längeren Schüssel. Aber woher könnt ihr dann wissen, dass der neue Schlüssel von uns ist und nicht vom Oberammergauer Staatsschutz, der auch mal RH-Schlüssel unterschreiben will? Nun, dafür folgen wir einem kleinen Protokoll, das wir auch euch für den Schlüsselwechsel ans Herz legen wollen – und ja, wenn ihr auch noch einen 1024 bit langen Schlüssel habt, dann wäre jetzt ein guter Zeitpunkt zum Üben.

## Schlüssel feilen

Wir beschreiben die Schritte beim Schlüsselumzug mit gnupg-Kommandozeilen (Version 2). Zwar geht das alles natürlich auch mit grafischen Programmen, aber die Kommandozeilen beschreiben klarer und knapper, was zu tun ist, und wir sind nicht von einem speziellen Programm abhängig. Bei Bedarf erklären die lokalen Nerd_innen bestimmt gerne den Umgang mit Kommandozeile und Editor – ihr könnt aber natürlich auch mit Maus und Menü eurer Schlüsselverwaltung nach Einträgen suchen, die ähnlich heißen.

Wir verwenden für die Beispiele die Schlüssel der Datenschutzgruppe; ihr müsst natürlich die Identitäten und Schlüssel-Ids an eure Situation anpassen.

*(1)* Der erste Schritt ist, einen neuen Schlüssel zu machen:

```
$ gpg --full-gen-key
```

Die Maschine fragt euch dann nach dem Schlüsseltyp (ihr wollt RSA und RSA), der Länge (ihr wollt 4096), dem Auslaufdatum des Schlüssels (dazu steht etwas im oben zitierten „Verschlüsseln mit Stil"; noch glauben wir, der Welt ist gedient, wenn ihr hier 0 schreibt) und schließlich der (ersten) Identität, für die dieser Schlüssel gelten soll, also etwa euer Gruppenname. Bei uns ist das „Datenschutzgruppe der Roten Hilfe Heidelberg" und im nächsten Schritt die Adresse datenschutzgruppe@rote-hilfe.de. Widersteht der Versuchung, einen Kommentar einzugeben[3] .

Wenn ihr schon PGP verwendet habt, ist euch die Wichtigkeit des nächsten Schritts, des Setzens einer Passphrase nämlich, ja bekannt. Wir merken nur nebenbei an, dass ein Zettel, der unmarkiert an einem halbwegs unverdächtigen Ort liegt, besser ist als ein toter Schlüssel, an dessen Passphrase sich niemand mehr erinnert.

Wie üblich solltet ihr dann viel Tippen und Maus bewegen, damit der Rechner genug Zufall bekommt, um euch einen Schlüssel zu machen. Wenn alles fertig ist, steht in der Ausgabe etwas wie:

```
pub   4096R/0x48475F525C0C5DB1 2016-04-16
    Key fingerprint = 4FD3 B3EE 7FCE 9...
uid                  Datenschutzgruppe...
sub   4096R/0x416FDB3D6E68FF12 2016-04-16
```

Wichtig für euch ist die Zahl hinter pub und der Schlüssellänge, also das, was mit 0x anfängt. Das ist die Schlüsselkennung. Kopiert sie an eine bequeme Stelle, ihr braucht sie im Folgenden noch öfter mal. Die Zahl kann bei euch auch nur halb so lang sein (was an eurer gnupg-Konfiguration liegt).

*(2)* Erzeugt euch als Nächstes ein Rückrufzertifikat des neuen Schlüssels:

```
$ gpg --gen-revoke 0x48475F525C0C5DB1 > r.txt
```

Ihr seht hier gleich mal den ersten Einsatz der Schlüsselkennung von eben. Das Kommando fragt euch nach einem Grund. Nehmt ruhig 1 (Schlüssel kompromittiert); in diesem Fall ist es gut, mit dem Schlimmsten zu rechnen. Nach diesem Kommando habt ihr in der Datei r.txt ein paar Byte, die, wenn irgendwer sie auf den Schlüsselserver hochlädt, euren Schlüssel (in guter Praxis) unbrauchbar macht. Das ist eine praktische Sache, wenn euer Schlüssel der Polizei in die Hand gefallen oder schlicht weg ist. Es ist aber doof, wenn Nazis das machen, um euch zu ärgern. Speichert das Rückrufzertifikat also sorgfältig, am besten nicht gerade auf eurer normalen Arbeitsmaschine.

*(3)* Damit Leute, die eurem alten Schlüssel trauen, auch eurem neuen trauen, unterschreibt euren alten mit eurem neuen Schlüssel. Schaut dazu erst nach, was die Kennung von eurem alten Schlüssel ist:

```
$ gpg --list-keys datenschutzgruppe
```

Ihr seht den alten und den neuen Schlüssel, und dann sagt ihr:

```
$ gpg --default-key 0xD1EAECCEF2BD132A \
  --sign-key 0x48475F525C0C5DB1
```

(der Backslash sagt: Gebt den Kram in einer Zeile ein, wir haben hier nur umgebrochen, damit die Zeilen nicht zu lang werden).

Das Ding mit 0xD1 davor ist die Kennung des *alten* Schlüssels, die im letzten Schritt rauskam. Die Passphrase, nach der gnupg dann fragt, ist auch die des

alten Schlüssels, denn er ist es ja, den ihr zum Unterschreiben benutzt.

*(4)* Nun muss noch irgendwie dafür gesorgt werden, dass der alte Schlüssel allmählich verschwindet. Die übliche Vorgehensweise ist, ihn noch etwas gelten zu lassen, damit eure Mailpartner_innen Zeit haben, sich auf den Übergang einzustellen. Dazu setzt ihr sein Ablaufdatum auf, sagen wir, ein Jahr in die Zukunft; für die Datenschutzgruppe lassen wir den Leuten sogar zwei Jahre Zeit. Das Setzen des Verfallsdatums lässt sich bequem in üblichen Schlüsselverwaltungen machen, oder über:

```
$ gpg --edit-key 0xD1EAECCEF2BD132A
```

Das führt auf eine eigene gpg-Kommandzeile, der ihr etwas wie `expire 1y` und `save` sagen könnt.

## Die Verkündung

Damit wärt ihr eigentlich fertig, bis auf das nebensächliche Detail, dass ihr der Welt eure Schlüsselmanipulationen mitteilen müsst. Einfach ist zunächst

*(5)* das Hochladen auf den Schlüsselserver. Wenn ihr das nicht aus eurer Schlüsselverwaltung tun wollt (alten *und* neuen Schlüssel!), könnt ihr auch einfach:

```
$ gpg --send-key 0xD1EAECCEF2BD132A \
  0x48475F525C0C5DB1
```

tippen (natürlich mit den Kennungen eurer Schlüssel).

*(6)* Jedoch ist ein Schlüsselwechsel für Leute, die euch verschlüsselte Mails schicken wollen, eine ziemlich große Sache – wie gesagt, es ist ja auch erstmal nicht klar, ob da nicht nur irgendwer versucht, anderen einen falschen Schlüssel unterzuschieben. Deshalb ist eine transparente Deklaration von dem, was da passiert, wichtig. Das übliche Mittel ist ein transition statement („Übergangserklärung").

In so einem Text sollte drinstehen

- Warum ihr einen neuen Schlüssel macht
- Ob und wie lange der alte noch vertrauenswürdig ist
- Schlüsselkennungen von altem und neuem Schlüssel sowie der Fingerprint zumindest des neuen.

Für die Datenschutzgruppe ist so ein Dokument unter https://datenschmutz.de/pgp-transition.txt zu finden.

Nun kann natürlich jede_r so eine Erklärung schreiben und ggf. sogar verbreiten. Um falsche Erklärungen zu verhindern, werden diese mit beiden Schlüsseln unterschrieben (das zeigt, dass zumindest mal kurz beide Schlüssel in einer Hand waren), und damit das Unterschreiben gut geht, sollte die Erklärung einfach eine Textdatei sein (also nicht mit Office-Programmen wie Word oder Libreoffice erstellt; unter Linux könnt ihr z.B. nano verwenden, mit Windows kommt das Programm Notepad mit). Wenn ihr eure Erklärung als pgp-transition.txt gespeichert habt, sagt ihr:

```
$ gpg --clearsign -u 0xD1EAECCEF2BD132A \
  -u 0x48475F525C0C5DB1 pgp-transition.txt
```

(wieder müsst ihr natürlich die Kennungen eurer Schlüssel statt der von uns eintragen). Das Kommando erzeugt eine Datei gpg-transition.txt.asc; das ist die, ihr verbreiten solltet – hier könnt ihr kreativ sein, aber ihr solltet sie wenigstens an eure Mailpartner_innen verschicken und, wenn ihr eine Webseite habt, auch dort unterbringen.

*(7)* Wenn der Schlüssel von einer Ortsgruppe der Roten Hilfe ist, schickt eure Erklärung bitte, möglichst in einem Anhang, über denn alle-Verteiler. Wenn wir euren alten Schlüssel signiert haben und innerhalb von einer Woche oder so niemand Einspruch gegen den Schlüsselwechsel erhebt, nehmen wir an, dass der Schlüsselwechsel im Interesse der gesamten Ortsgruppe geschehen ist und signieren, wenn die Übergangserklärung in Ordnung ist, ohne Weiteres auch den neuen Schlüssel.

*(8)* Wenn ihr selbst Schlüssel signiert habt, wäre jetzt ein guter Zeitpunkt, nachzusehen, welche von denen ihr auch mit dem neuen Schlüssel signieren wollt. Wie pingelig ihr da seid, ist etwas Geschmackssache. Als Leitplanken: Es dürfte unstrittig sein, dass ihr einen Schlüssel, den ihr gerade in der Vorwoche geprüft habt, bedenkenlos erneut unterschreiben könnt. Einen Schlüssel, mit dem ihr vor fünf Jahren das letzte Mal was gemacht habt, solltet ihr wohl nicht neu unterschreiben.

## Was tun wenns brennt?

Wenn euer Schlüssel tatsächlich kompromittiert wurde, also etwa der Polizei in die Hände gefallen ist oder von Menschen genutzt wurde, denen ihr nicht mehr traut, ist der erste Schritt, ihn zurückzurufen. Dabei kommt das Rückrufzertifikat ins Spiel, von dem oben die Rede war – das solltet ihr jetzt nämlich auf die Schlüsselserver hochladen[4]. Im Effekt werden Leute,

die ihre Schlüssel mit den Schlüsselservern synchronisieren, den kompromittierten Schlüssel nicht mehr verwenden.

Weil (gegenwärtig) die wenigsten Leute ihre Schlüssel regelmäßig mit den Schlüsselservern abgleichen, solltet ihr aber Leuten, die euren Schlüssel haben, per Mail Bescheid sagen, am besten gleich mit dem Vermailen eurer Übergangserklärung. Die sollte dann sehr deutlich sagen, dass der alte Schlüssel kompromittiert ist und damit nicht mehr benutzt werden darf. Entsprechend könnt ihr euch die Signatur des neuen Schlüssels und der Übergangserklärung mit dem alten Schlüssel schenken.

Ähnlich, wenn auch nicht ganz so drastisch, könnt ihr vorgehen, wenn ihr beispielsweise eure Passphrase vergessen habt oder der Schlüssel bei einer Computerkatastrophe ohne Backup kaputt gegangen ist. Speziell in solchen Fällen zahlt es sich aus, wenn es mindestens zwei (vertrauenswürdige) Personen gibt, die eine Kopie des Rückrufzertifikats haben.

Um versöhnlich zu schließen: PGP ist auch mit schlechtem Schlüsselmanagement noch besser ist als gar keine Verschlüsselung oder Mumpitz wie de-Mail. Wer sich von dem ganzen Gerede von signierten Schlüsseln abgeschreckt fühlt, kann es immer noch ignorieren. In den sieben Jahren, die seit unserer Prognose, die Staatsgewalt werde noch für eine Weile nicht mit unseren Schlüsseln rumspielen, vergangen sind, hat sie es nach unserer Kenntnis in der Tat nicht getan.

Wer aber 1024D oder 1024R-Schlüssel hat und nicht ganz verschreckt ist von dem Zeug: Probierts mal aus. Für die nächsten sieben Jahre möchten wir nämlich im Hinblick auf die Staatsgewalt nichts versprechen.

Datenschutzgruppe der Roten Hilfe Heidelberg

Kontakt und Artikel-Archiv: https://datenschmutz.de

PGP Fingerprint **(neuer Schlüssel)**: 4FD3 B3EE 7FCE 9FFD EC75 CAF9 4847 5F52 5C0C 5DB1

[1] https://datenschmutz.de/gc/html/pgppraxis.html

[2] https://datenschmutz.de/gc/html/weboftrust.html

[3] https://www.debian-administration.org/users/dkg/weblog/97

[4] Ein Web-Formular, auf dem ihr das machen könnt, ist bei https://sks-keyservers.net/i/