Verschlüsselung

21.1.2015

Hier möchte ich meine Erfahrungen berichten und dokumentieren:

Mein öffentlicher PGP-Schlüssel für eMail: Schlüssel-Datei
Oder auf dem Schlüsselserver http://keys.gnupg.org

eMail und PGP

Standardmäßig ist die Kommunikation per eMail immer unverschlüsselt. D.h. jeder, der auf der Leitung vom Sender bis zum Empfänger mithorchen kann, kann den Inhalt auch lesen.
Hier ein nettes Video dazu; Der digitale Briefumschlag

Abhilfe schafft Verschlüsselung mit **Pretty good Privacy (PGP)** bzw. die freie Variante davon vom GNU-Projekt namen **GNU Privacy Guard (GPG)**.

Programme

Ich verwende folgende Programme:

- **gpg** (GNU privacy guard) als Hintergrund-Programm; ist auf *Kubuntu (Linux)* standardmäßig dabei
- **gpa** (GNU privacy assistant) zum Verwalten und Erzeugen der Schlüssel
- **Thunderbird**, mein eMail-Programm mit Adressbuch und Kalender
- **Enigmail**, ein Plugin für Thunderbird, das die Verschlüsselung innerhalb Thunderbird umsetzt.

Für *Windows* gibts anstatt gpg und gpa das Programm gpg4win

Installation

Die Installation ist fast selbsterklärend.

1. Schlüsselpaar mit gpa erzeugen (min RSA 2048, 1024 ist zu schwach)
2. Enigmail in Thunderbird installieren (über Tools → Add-Ons)
3. Enigmail einrichten
4. fertig.

PGP, HTML, PGP/MIME und S/MIME

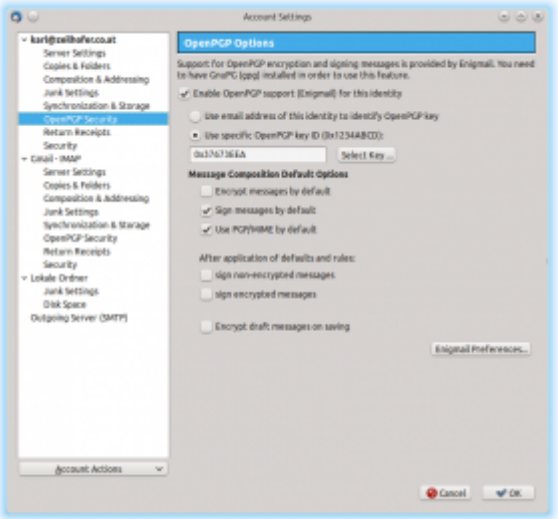
23.1.2015

Enigmail deaktiviert das schreiben und lesen von HTML-formattierten eMails.
Das wär aber ganz nett, wenn es doch funktionieren würde, denn eMails mit nur reinem Text haben

dann keine Möglichkeit Bilder einzubetten, Text fett zu schreiben, Farben zu verwenden und noch vieles mehr.

Das Thema ist für mich noch eher undurchsichtig. Angeblich verträgt sich PGP mit **HTML** so direkt nicht. Ein Ausweg ist, **PGP/MIME**-formatierte eMails zu versenden. MIME ist eine Richtlinie (oder ein Standard?), wie solch eine eMail im Rohformat auszusehen hat. Das größte Problem hiermit ist angeblich, dass ein Empfänger mit **Microsoft Outlook** die eMail nicht öffnen kann, weil sich Microsoft scheinbar dagegen streubt, PGP/MIME zu unterstützen.
Ich persönlich unterstütze Microsoft nicht, daher ist mir das erst einmal egal. Wer meine eMails nicht lesen kann, soll sich dann halt melden. Der bekommt im Adressbuch einen vermerk, dass eMails an ihn dann nicht mehr signiert und verschlüsselt werden oder halt nur als reine Text-eMail erstellt werden.

PGP/MIME aktiviert man folgendermaßen: Menu → Edit → Account Settings –> select account → OpenPGP Security Hier kann man dann den Haken bei „Use PGP/MIME by default“ setzen. Dadurch benachteiligt man wie oben beschrieben Empfänger mit Microsoft Outlook.



S/MIME ist scheinbar ein neuerer Standard, der auch von Microsoft unterstützt wird. Ein wenig unbehagen habe ich dabei, wenn man sich das X.509-Zertifikat dann erst wieder von einer Zertifizierungsstelle ausstellen lassen muss. Hier wird erklärt, wie man sich selbst-signierte Schlüssel mit OpenSSL erzeugt: Youtube
Weitere Infos dazu kommen noch…

Key-Server

Damit der öffentliche Schlüssel für die Empfänger zur Verfügung steht, sollte man seinen öffentlichen Schlüssel auf einen öffentlichen Schlüsselserver (siehe Wikipedia) hochladen. Auf Linux kann man das mit dem gpa machen.
Mein key liegt nun auf dem Server keys.gnupg.net, wobei er mit der Schlüssel-ID im hexadecimalen Format 0x37673EEA zu finden sein sollte.

Hinweise

Durch PGP wird nur der Inhalt der eMail verschlüsselt, also auch Anhänge. Offengelegt bleibt immer

noch der Betreff, der Absender und Empfänger der eMail. Dies wird auch Metadaten genannt, d.h. wer wann mit wem worüber (nur bei aussagekräfigem Betreff) kommuniziert hat. Das sind oft weitaus interessantere Daten, als der Inhalt der Nachrcht selbst. Dadurch kann man aussagekräftige Profile über Personen erstellen, das was die NSA auch überall macht.
Nichtsdestotrotz wollen wir ersteinmal mit der reinen Verschlüsselung und Signierung anfangen.

Jedes versendete eMail wird auch automatisch **signiert**. D.h. der Empfänger hat die Möglichkeit zu prüfen, ob wirklich wir die Sender der eMail sind, oder ob sich jemand als Karl Zeilhofer in meinem Fall ausgibt.

Das erzeugen des Schlüsselpaares benötigt ein **Passwort (passphrase)**. Dieses wird beim Versenden und beim Entschlüsseln von eMails immer benötigt. Hat man kein Passwort angegeben, hat jeder, der die Datei mit dem privaten Schlüssel hat, die gleichen Möglichkeiten wie man selbst.

Von dem Schlüsselpaar sollte man unbedingt eine Sicherungskopie mit gpa machen. Denn ohne dem privaten Schlüssel kann man seine eMails nicht mehr entschlüsseln!

Ein Schlüssel hat einen **Fingerabdruck (engl. fingerprint)**. Die ist eine Zahl, die aus dem Schlüssel abgeleitet wird. Hiermit können die beiden Kommunikaitonspartner überprüfen, ob sie wirklich die gleichen Schlüssel verwenden, und sichergehen, dass niemand einen gefälschten Schlüssel auf dem Übertragungsweg eingeschleust hat. Der Fingerprint meines Schlüssels ist:

```
4A75 AB41 0D9B 0BEC F751  38C2 8469 E901 3767 3EEA
```

gpa und Error: "unknown certificate"

gpa mit sudo starten, dann funktioniert es.
Dies führt jedoch zu Veränderungen der Schreib- und Leserechte in ~/.gnupg/
Hier hilft dann:

```
sudo chown karl:karl *
chmod 600 *
```

Nun läuft gpa auch wieder einwandfrei…

PGP auf Android

Nachdem die Verschlüsselung nun auf dem PC bereits funktioniert, macht es in heutiger Zeit auch Sinn, dies auch auf dem Smartphone einzurichten.

Programme

- **K-9 Mail** als eMail Programm (open source)
- **AGP** für die Schlüsselverwaltung

Installation

1. Zuerst AGP installieren.
2. Schlüssel-Backup vom PC auf SD-Karte im Handy übertragen. In gpa Schlüssel auswählen und im Menü „Keys → Backup“ verwenden. \\Wichtig: Nicht die Funktion „Export Key“ verwenden, denn hiermit würde nur der öffentliche Schlüssel abgespeichert werden.
3. Diese Datei lädt man dann in AGP mit „Import Key“. \\Hinweise: wenn man noch das USB-Kabel am Handy stecken hat, kann man auf die SD-Karte nicht zugreifen :)
4. K-9 Mail installieren und einrichten
5. Schlüsseldatei von der SD-Karte wieder löschen
6. fertig.

Hinweise

Das Passwort für den Schlüssel sollte Handy-tauglich sein. Viele Ziffern und Sonderzeichen würd ich eher vermeiden. Das Passwort kann man im Nachhinein auch verändern.

Passwort-Tresor mit KeePass

Für Verschlüsselung sind meist auch Passwörter notwendig. Überall das gleiche Passwort zu verwenden ist eine schlechte Idee. Denn wird einer der verwendeten Dienste gehackt, so bekommt der Angreifer Zugang zu allen Diensten, die man sonst noch so verwendet.

Vermutlich kann man sich aber zig verschiedene Passwörter nicht merken. Meiner Erfahrung kann man sich nicht einmal alle Seiten merken, bei denen man registriert ist.

Die Lösung ist ein Passwort-Tresor. Dieser ist nur zugänglich über ein Passwort und/oder eine Schlüssel-Datei. Eine mögliche Variante, die ich nun seit über einem Jahr in täglicher Verwendung habe, ist **KeePass**. KeePass gibt es für alle üblichen PC und Smartphone-Betriebssysteme. Weitere Infos gibt es auf Wikipedia: englisch oder deutsch

KeePass verwendet eine verschlüsselte Datenbank, die **in einer einzigen Datei** abgespeichert wird. D.h. überall, wo man diese Datei hat, und wo KeePass läuft, kann man all seine Passwörter abrufen. Weiters kann man darin auch gescannte sensible Dokumente wie Führerschein, Reisepass, Staatsbürgerschaftnachweis, TAN-Listen und ähnliches verwahren.

Meine Datenbank hat mittlerweile fast 130 Einträge. Das kann man sich beim besten Willen nicht alles merken. Vielen Dank hier an Christian M. Schmid, der mich in seinem Blog-Artikel darauf aufmerksam gemacht hatte

software, deutsch, email, pgp, gpg, security, android, artikel