# Dokumente digital signieren unter Linux

*OpenOffice.org*

# Copyright

## Copyright und Lizenzen



Der Titel der Originaldokumentation ist „Dokumente digital signieren unter Linux“.

Der/die ursprünglichen Autoren der Originaldokumentation sind im Abschnitt „Autoren“ aufgeführt und können entweder unter authors@user-faq.openoffice.org oder bei Fragen/Anmerkungen zur Übersetzung unter SimonAW@openoffice.org (IRC-Nickname: SimonAW) kontaktiert werden.

Personen, die das Dokument in irgendeiner Weise nach dem unten angegebenen Veröffentlichungsdatum verändern, werden im Abschnitt „Beitragende“ mitsamt Datum der Veränderung aufgeführt.

| **Autoren** | **Beitragende** |
|---|---|
| Simon A. Wilper | |

## Veröffentlichung und Softwareversion

Das Dokument basiert auf der Version 2.1 von OpenOffice.org. Dieses Dokument wurde am 18. Februar 2008 veröffentlicht.

# Inhaltsverzeichnis

# Übersicht

Hier erfahren Sie, wie Sie mit Hilfe von Programmen, die schon bei gängigen Linuxdistributionen vorhanden sind, ein Zertifikat anlegen, mit dem man dann OpenOffice.org-Dokumente digital signieren kann.

Es wird davon ausgegangen, dass der Anwender mit der Shell und der verwendeten Linux-Distribution vertraut ist.

# Voraussetzungen

Um digitales Signieren von OpenOffice.org Dokumenten zu unterstützen, sind unter Linux folgende weitere Programme notwendig:

- **OpenSSL**
  stellt die erforderlichen Verschlüsselungsalgorithmen zur Verfügung, mit denen zunächst ein RSA-Schlüssel generiert wird und folgend das Zertifikat.
- **Mozilla Thunderbird**
  oder Firefox ist notwendig um das Zertifikat in deren cert8.db zu importieren. OpenOffice.org greift beim Signieren darauf zu.

| Anmerkung | Wenn Firefox und Thunderbird installiert sind, erkennt OpenOffice.org nur die cert8.db von Thunderbird. |
|---|---|

# Erstellen des Zertifikats

Es gibt zwei grundlegende Wege, das Zertifikat zu erstellen. Man generiert ein sogenanntes Certificate Request, schickt es einer Certificate Authority (CA) und bekommt ein signiertes Zertifikat zurück, mit dem man dann fortfahren könnte.

Da dies jedoch in der Regel (viel) Geld kostet - außer z.B. bei CACert (cacert.org) - entscheiden wir uns für die Variante des selbst-signierten Zertifikats.

Zur Erstellung des Zertifikats benutzt man in der Regel das Programm `openssl` aus der dem Paket `openssl`, welches ausschließlich auf der Shell arbeitet. Darauf aufsetzend existiert TinyCA (tinyca.sm-zone.net), das ein GTK-Frontend zur einfacheren Bedienung bietet.

Auf einer Shell (in den Beispielen wird von Bash ausgegangen) führen Sie zunächst folgenden Befehl aus:

```
$ openssl genrsa -des3 -out privkey.pem 2048
```

| *Parameter* | *Eläuterungen* |
|---|---|
| `genrsa` | RSA-Schlüssel erzeugen<br>Da das Verfahren der Zertifikate auf einem Public-Key-Verfahren beruht, wird erst der private Schlüssel erzeugt von dem dann der öffentliche erstellt wird. |
| `-des3` | Anforderung einer Passphrase, der für die Verschlüsselung benutzt wird. |
| `-out privkey.pem` | Ausgabe in Datei privkey.pem |
| `2048` | Länge des Modulus |

Wenn der Befehl abgesetzt wird, sollten in etwa folgende Zeilen erscheinen:

```
Generating RSA private key, 2048 bit long modulus
.................................+++
...........+++
e is 65537 (0x10001)
Enter pass phrase for privkey.pem:
Verifying - Enter pass phrase for privkey.pem:
```

| Anmerkung | Wenn Sie das Passwort eingeben, werden keine Echozeichen angezeigt. |
|---|---|

Der Schlüssel - hier im Beispiel privkey.pem - sollte in etwa so aussehen (gekürzte Fassung):

```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,75BEF0DBDEB1FEE9

tHZsHxvMuqw8EzK0Ud65cicQ4960RV9J1EsDZ4wJKCxwTlFQS1/44ayj6Tf1eBRW
.
.
.
0O5GqXP/1FVzUrhHEci0T33PVgQLwgw2WorMrHTeMTFLfXq2dUpetPwx7vcbAS33
-----END RSA PRIVATE KEY-----
```

Als nächstes ist das Zertifikat zu erstellen mittels:

```
$ openssl req -new -x509 -key privkey.pem -out cacert.pem -days 31
```

| ***Parameter*** | ***Eläuterungen*** |
|---|---|
| `req` | X.509 Certificate Signing Request (CSR) Management |
| `-new`<br>`-x509` | Erstellen des X.509-Zertifikats<br>X.509 ist ein ITU-T-Standard für eine Public-Key-Infrastruktur und derzeit der wichtigste Standard für digitale Zertifikate. Die aktuelle Version ist X.509v3. [Q: Wikipedia.de] |
| `-key privkey.pem` | der zu benutzende private Schlüssel |
| `-out cacert.pem` | Ausgabedatei |
| `-days 31` | Gültigkeitszeitraum von 31 Tagen |

Erst wird nach dem Passwort für den privaten Schlüssel gefragt:

```
Enter pass phrase for privkey.pem:
```

Darauffolgend werden Sie nach einigen standortbezogenen und persönlichen Informationen über sich gefragt, die in das Zertifikat einfließen:

```
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a
DN.
```

```
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:DE
State or Province Name (full name) [Some-State]:Nordrhein Westfalen
Locality Name (eg, city) []:Hamm
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
OpenOffice.org
Organizational Unit Name (eg, section) []:Research and Development
Common Name (eg, YOUR name) []:Simon A. Wilper
Email Address []:simonaw@openoffice.org
```

Das Zertifikat sollte danach in der Datei `cacert.pem` vorliegen.

Der letzte Schritt besteht darin, das Zertifikat in das PKCS#12-Format zu überführen, da Mozilla Firefox und Thunderbird nur dieses Format importieren können:

```
$ openssl pkcs12 -export -inkey privkey.pem -in cacert.pem -out
cacert.p12
```

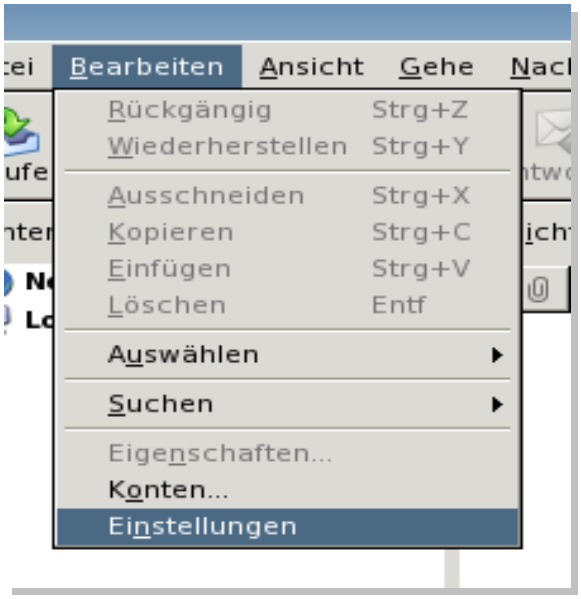
| ***Parameter*** | ***Eläuterungen*** |
|---|---|
| `pkcs12` | pkcs12-Modul aufrufen |
| `-export` | Aktion „Export“ |
| `-inkey privkey.pem` | Eingabeschlüssel |
| `-in cacert.pem` | Eingabezertifikat |
| `-out cacert.p12` | Ausgabezertifikat |

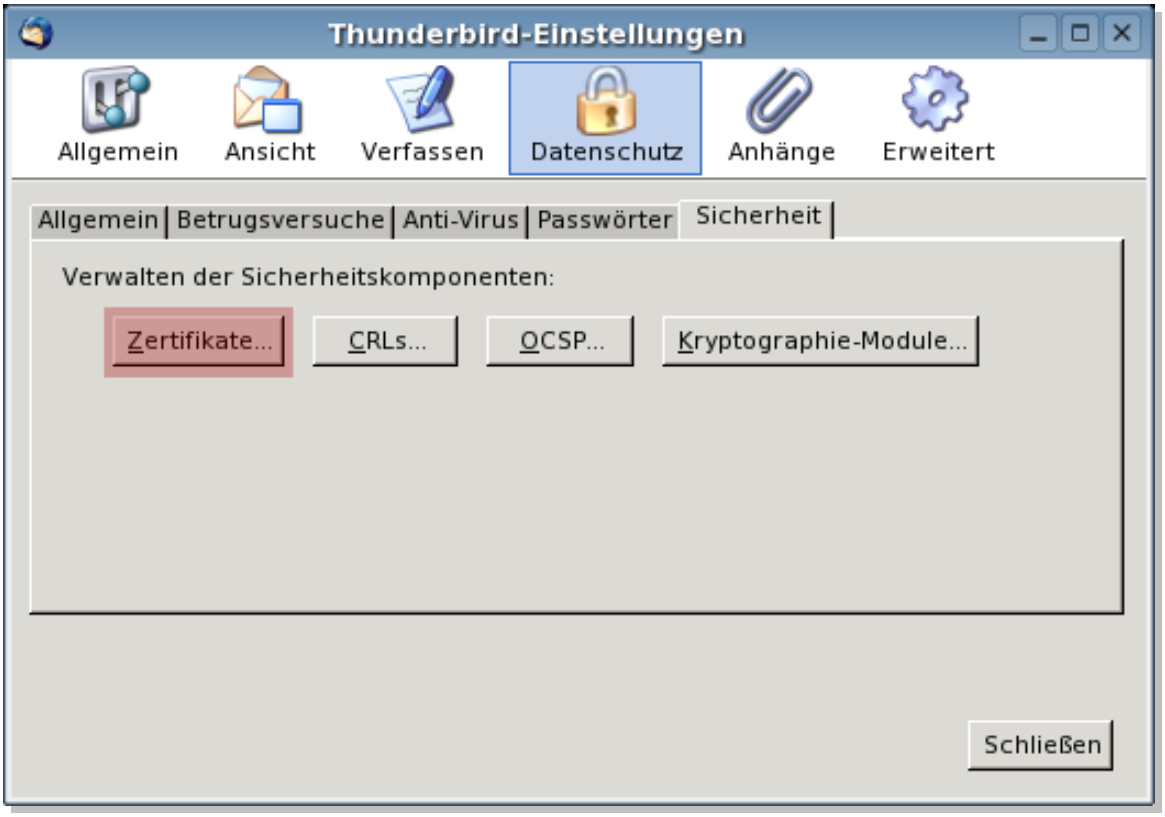
Hier wird zunächst ein weiteres Mal das Passwort für den privaten Schlüssel angefordert und ein zweites Export-Passwort, das zweimal eingegeben wird:

```
Enter pass phrase for privkey.pem:
Enter Export Password:
Verifying - Enter Export Password:
```

Es sollte nun die Datei cacert.p12 vorliegen.

# Import der PKCS-Datei in Thunderbird

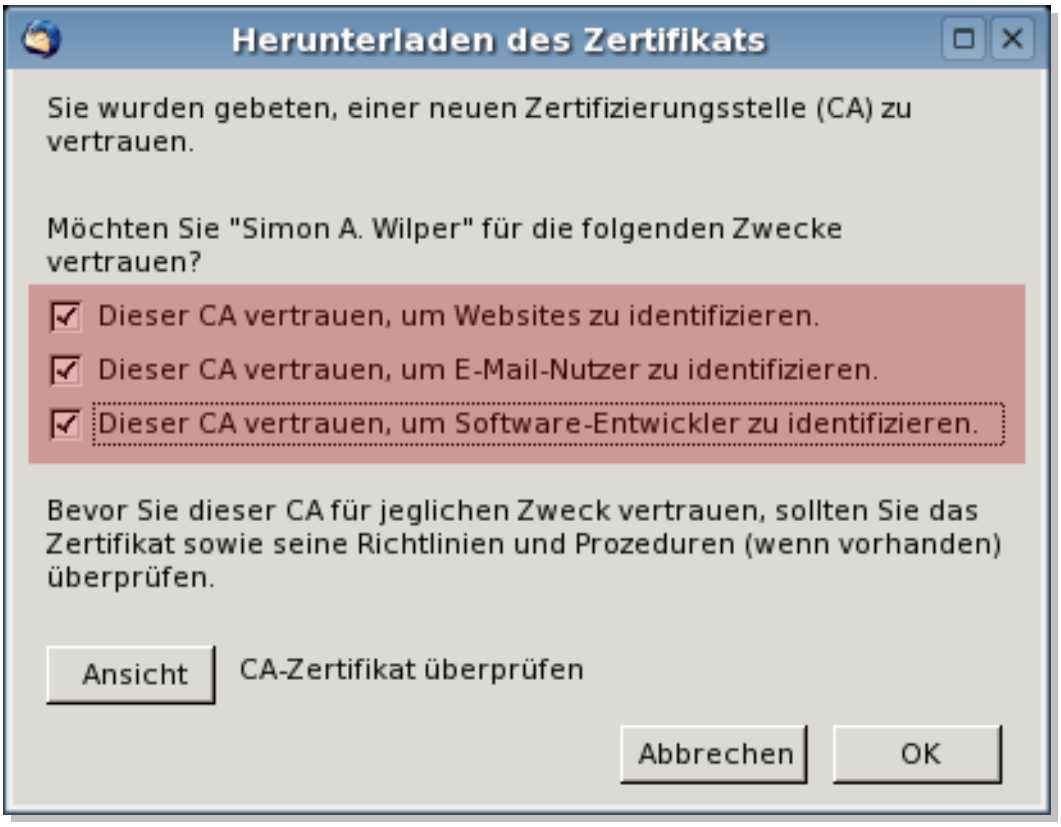



Starten Sie nun Mozilla Thunderbird und öffnen Sie den Optionsdialog über *Bearbeiten - Einstellungen.*

Klicken Sie auf die Kategorie *Datenschutz,* Tab *Sicherheit*. Nach Klick auf den Knopf *Zertifikate* erscheint folgendes Dialogfenster:


Der Import läuft in zwei Schritten ab. Zunächst importieren Sie die cacert.pem, die Sie zuvor angelegt haben in die Zertifizierungsstellen (Authorities). Damit ist sichergestellt, dass das Zertifikat gültig ist.




Aktivieren Sie alle Checkboxen.

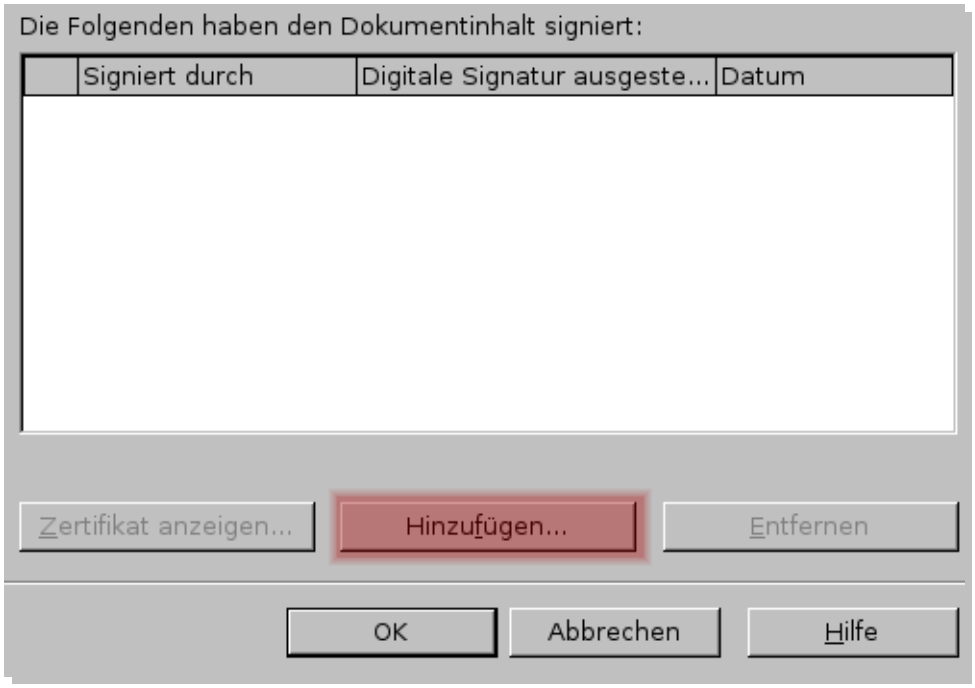
Importieren Sie nun als zweiten Schritt die PKCS#12-Datei mittels Importieren in die Gruppe *Ihre Zertifiakte*. Es wird nach dem Master-Password des Software Security Devices gefragt. Hierbei handelt es sich um das Passwort, das bei der Generierung des privaten Schlüssels angegeben wurde. Nach dessen Eingabe wird das zweite

Passwort angefordert, jenes, mit dem der Schlüssel in das PKCS#12-Format exportiert wurde.

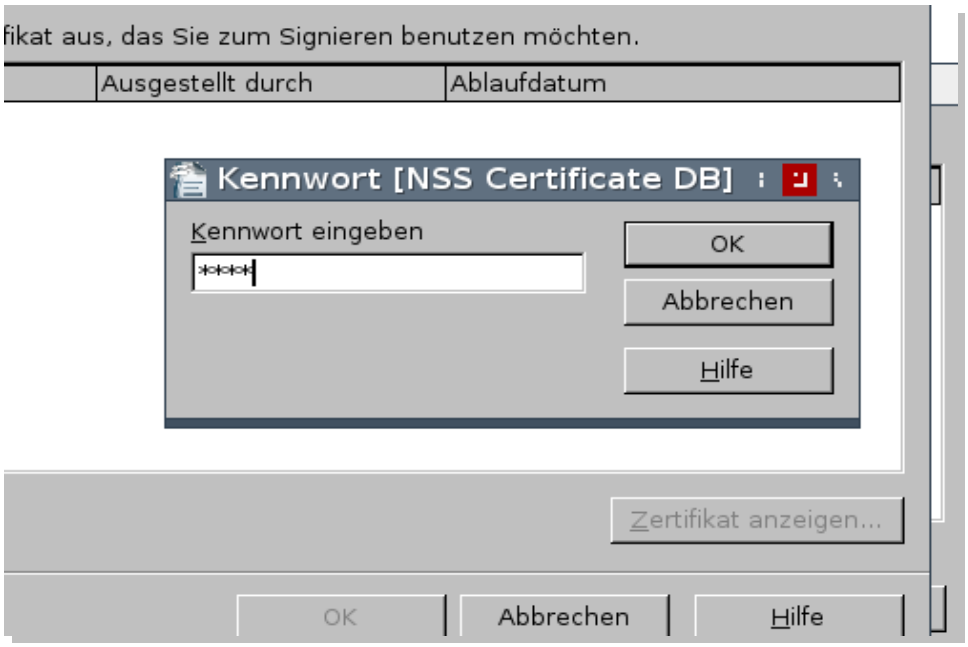

Somit sollte eine Meldung erscheinen, die das erfolgreiche Importieren des Zertifikats bescheinigt. Mit Klick auf Anzeigen kann man sich noch einmal vom Zustand und der Richtigkeit aller Daten überzeugen, bevor man ein OpenOffice.org-Dokument damit signiert.

# Signieren eines Textdokuments

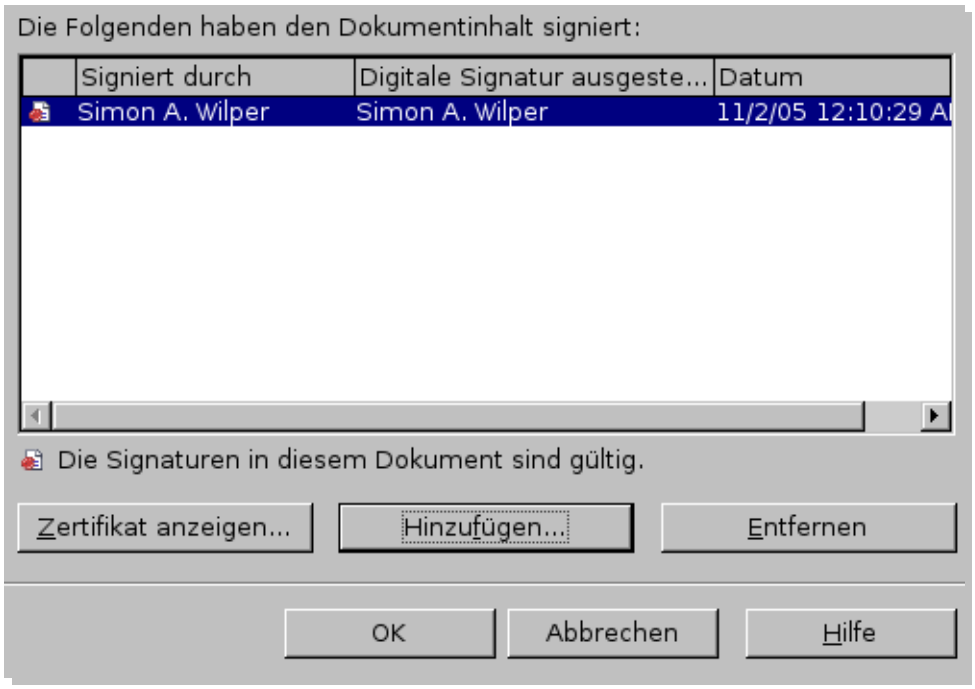
Starten Sie jetzt OpenOffice.org, erstellen Sie ein neues Textdokument und speichern es. Rufen Sie den Dialog Digitale Signaturen mittels Datei - Digitale Signaturen auf und fügen ein Zertifikat hinzu.



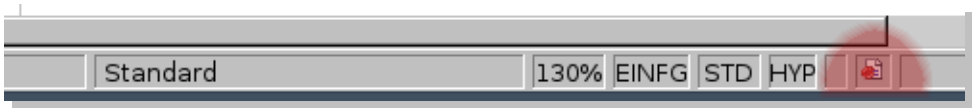
Durch Klicken auf Hinzufügen werden zwei neue Fenster geöffnet:




Es wird wiederum nach dem Passwort gefragt, mit dem der private Schlüssel verschlüsselt wurde. Ein weiteres Fenster öffnet sich, das die verfügbaren Zertifikate auflistet. Wählen Sie das entsprechende aus und bestätigen Sie. Das Fenster Digitale Signaturen sollte nun ein Element enthalten:




Klicken Sie auf OK. In der Statuszeile sollte nun ein rotes Siegel-Symbol erscheinen:

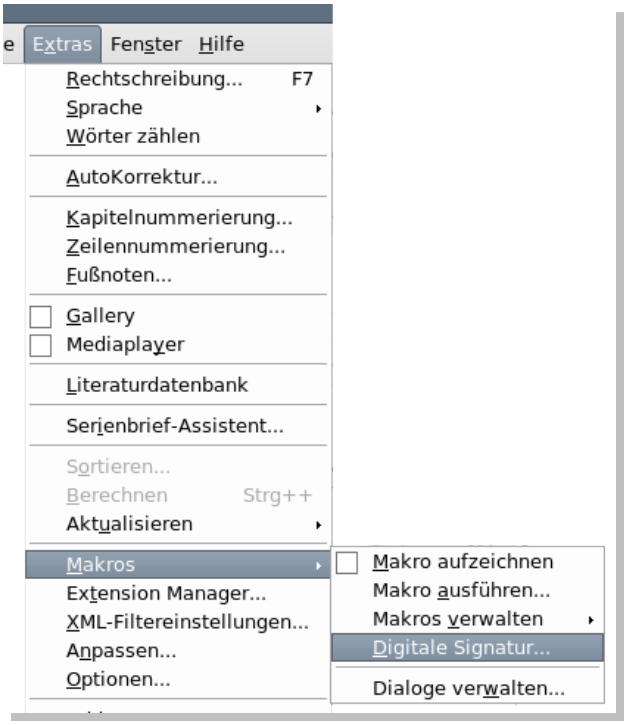



# Signieren von Makros

In der Standardeinstellung von OpenOffice.org werden Makros aus nicht vertrauenswürdigen Quellen standardmäßig nicht ausgeführt.

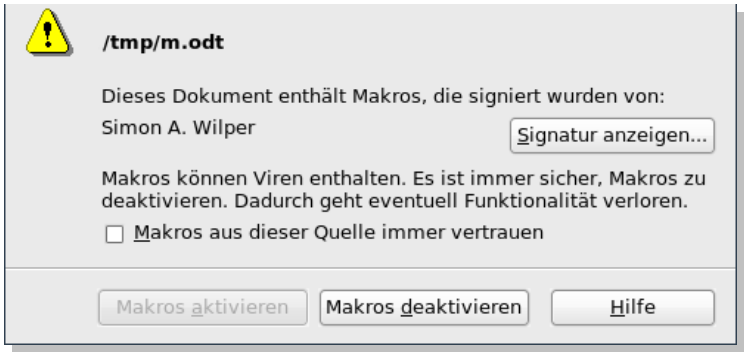
Somit hilft entweder das Hinzufügen des Verzeichnisses zu den vertrauenswürdigen Quellen oder die Sicherheitsstufe herabzusetzen, was ein Sicherheitsrisiko darstellt.

Der optimale Weg wäre, das Makro mit einer digitalen Signatur auszustatten, damit OpenOffice.org die Ausführung sofort gewährt.



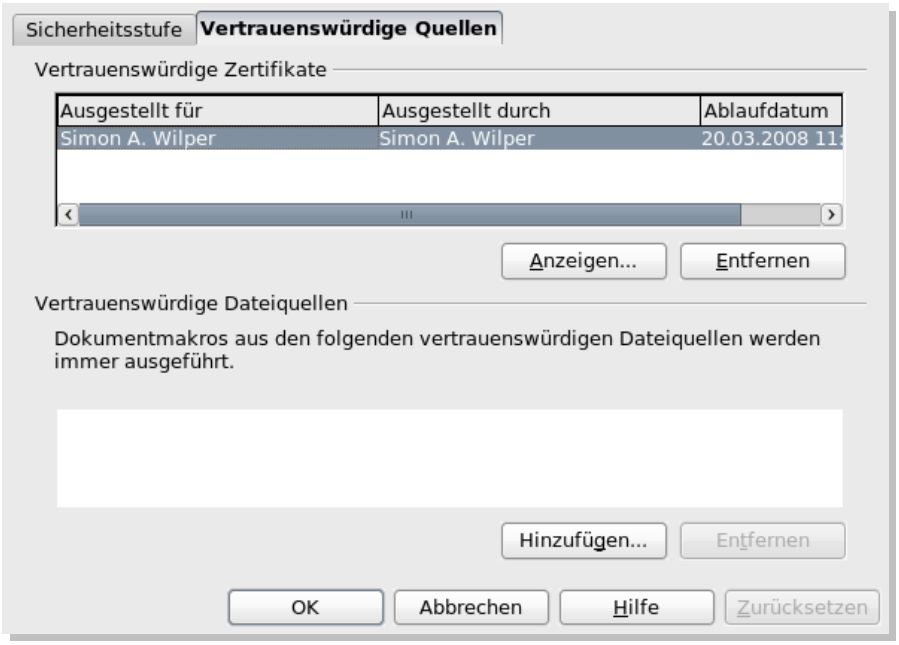

Es öffnet sich der selbe Dialog zur Zertifikatauswahl. Nachdem Sie das Zertifikat hinzugefügt haben, sind alle Zertifkate signiert.

Der Anwender wird dennoch mit einem Bestätigungsdialog konfrontiert, wenn er Ihr Dokument öffnet:




Der Benutzer muss die Checkbox „Makros aus dieser Quelle immer vertrauen“ aktivieren, bevor er über die Schaltfläche „Makros aktivieren“ fortfahren kann.

Das Zertifikat wurde in die Liste der vertrauenswürdigen Quellen aufgenommen:




Ab sofort werden Makros sofort ausgeführt, die mit diesem Zertifikat signiert wurden und solange das Zertifikat gültig ist.