Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Eidgenössisches Finanzdepartement EFD

Bundesamt für Informatik und Telekommunikation BIT
Lösungszentrum

Richard Schaller 15. April 2009

e-dec

Entschlüsselung und Signaturprüfung mit dem OpenSSL Toolkit für Emailverkehr

Projektname:	**e-dec**
Projektnummer	**LZBE106-00109**
Version:	**1.3**
Datum:	**2009-04-15**

Status	in Arbeit	in Prüfung	genehmigt zur Nutzung
	☐	☐	☑

Beteiligter Personenkreis	
Autoren:	Richard Schaller (rsc) Matthias Rüedlinger (mru)
Genehmigung:	Projektteam e-dec IDEE
Benützer/Anwender:	EZV, PL
zur Information/Kenntnis:	EZV

Änderungskontrolle, Prüfung, Genehmigung			
Wann	**Version**	**Wer**	**Beschreibung**
2008-12-02	1.1	rsc	Neues Format; Bugfix für OpenSSL aktualisiert
2009-03-31	1.2	mru	Kapitel 1.2 und 1.4 (Verschlüsselung) hinzugefügt
2009-04-15	1.3	mru	Referenzen hinzugefügt

Inhaltsverzeichnis

1 Vorgehen

1.1 Allgemeines

E-dec erzeugt verschlüsselte/signierte Mails mittels IAIK JCE Tookit (http://jce.iaik.tugraz.at/products/01%5Fjce/).

Mit der Distribution von OpenSSL 0.9.8i für Windows können die nachfolgenden Schritte durchgeführt werden (http://www.openssl.org/).

Es werden folgende Zertifikate benötigt:

- Privates Spediteurzertifikat
- Öffentliches Zertifikat für e-dec für die entsprechende Umgebung (Test oder Produktion)

1.2 Entschlüsseln und Signaturprüfung mit dem OpenSSL Toolkit

1. Erstellen eines OpenSSL PEM Zertifikates

```
$ openssl pkcs12 -in test_spediteur@ezv.admin.ch.p12 -out
test_spediteur\@ezv.admin.ch.pem
Enter Import Password:
MAC verified OK
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
```

2. Speichern des Sourcecodes der edec Response Message in raw_mail.txt

3. Entschlüsseln

```
$ openssl smime -decrypt -in raw_mail.txt -out mail_decrypted_b64.txt
     -recip test_spediteur@ezv.admin.ch.pem
```

4. Umwandlung base64 zu Klartext

```
$ openssl smime -verify -noverify -in mail_decrypted_b64.txt
     -out response.txt
Verification successful
```

Bemerkung zu 1), 3):

test_spediteur@ezv.admin.ch.p12 sollte mit dem jeweiligen Spediteurzertifikat ersetzt werden.

Bemerkung zu 4):

Die Signatur wird durch openssl smime -verify -noverify -in **raw_mail.txt** -out **response.txt** bereits verifiziert. Es sollte auch eine Meldung wie "Verification successful" ausgegeben wer-

den. Das -noverify bewrikt, dass die Zertifikate nicht verifiziert werden. Wenn -noverify weggelassen wird, werden auch die Zertifikate verifiziert. Dazu ist es dann jedoch auch notwendig, einen "Trust-Anchor" anzugeben, z.B. über die -CAfile Option:

```
$ openssl smime -verify -in response_decrypted.b64 -out response.txt
    --CAfile caCert.pem
```

1.3 Singnieren und Verschlüsseln mit dem OpenSSL Toolkit

1. Erstellen eines OpenSSL PEM Zertifikates

   ```
   $ openssl pkcs7 -inform DER -in zoll_zertifikat.p7b -print_certs
       -out e-dec.pem
   ```

2. Signieren der Nachricht mit dem Spediteurzertifikat

   ```
   $ openssl smime -sign -in message.txt -text -out signed.txt
       -signer test_spediteur@ezv.admin.ch.pem
   ```

3. Die signierte Nachricht verschlüsseln

   ```
   $ openssl smime -encrypt -in signed.txt -out encrypted.txt
       -from test_spediteur@ezv.admin.ch
       -to cutoms_declaration_a@edec.ezv.admin.ch
       -subject "Signed and Encrypted message" -des3 e-dec.pem
   ```

Bemerkung zu 1):

test_spediteur@ezv.admin.ch.pem sollte mit dem jeweiligen PEM Spediteurzertifikat ersetzt werden.

Bemerkung zu 2):

e-dec.pem sollte mit dem PEM Zertifikat der verwendeten e-dec Umgebung (Test / Produktion) ersetzt werden

Bemerkung zu 3):

Befinden sich mehre Zertifikate in der PEM Datei wird von OpenSSL das erste Zertifikat verwendet und die restlichen Zertifikate ignoriert.

1.4 Verschlüsselung/Entschlüsseln mit OpenSSL (von CSF/Wegener)

1.4.1 Verschlüsseln und signieren

```
edecSignerZert=$HOME/certs/zoll90.cert
edecSignerKey=$HOME/certs/zoll90.key

openssl smime -sign -in $Datei.64 -signer $edecSignerZert -inkey $edecSignerKey \
| openssl smime -encrypt  \
        -from $AbsMail -to $EmpfMail \
        -subject "ZOLL90XML_TEST_$Datei" -des3 $PublicCert \
| /usr/lib/sendmail $EmpfMail
```

1.4.2 Entschlüsseln und verifizieren

```
edecSignerZert=$HOME/certs/edec.cert
edecSignerKey=$HOME/certs/edec.key

openssl smime -decrypt -in $DAT -out mail_decrypted_b64.txt -recip $edecSignerZert -inkey
$edecSignerKey
openssl smime -verify -noverify -in mail_decrypted_b64.txt -out mail.txt
```

1.5 Probleme beim Verifizieren der Signatur mit OpenSSL

Fehler:

```
Error reading S/MIME message
10907:error:0D06B078:asn1 encoding routines:ASN1_get_object:header too
long:asn1_lib.c:140:
10907:error:21078082:PKCS7 routines:B64_READ_PKCS7:decode er-
ror:pk7_mime.c:142:
10907:error:2107A08B:PKCS7 routines:SMIME_read_PKCS7:pkcs7 parse er-
ror:pk7_mime.c:299:
```

Lösung:

Das Kunden Service Center (e-dec.helpdesk@ezv.admin.ch) kann in dem Fall kontaktiert werden, welche Ihnen eine Script oder eine Javaapplikation zur Problembehebung zur Verfügung stellt.

2 Referenzen

Certificate Management with OpenSSL	http://gagravarr.org/writing/openssl-certs/general.shtml
OpenSSL Online Dokumentation	http://openssl.org/docs/
OpenSSL SMIME	http://openssl.org/docs/apps/smime.html
OpenSSL for Windows	http://www.slproweb.com/products/Win32OpenSSL.html