Best Practice zur Security Schnittstelle

Um den Empfehlungen des BSI zu entsprechen, wird in einer Migration auf eine Schlüssellänge von 4096 Bit und bei Signaturverfahren auf den Algorithmus RSASSA-PSS sowie bei Verschlüsselungsverfahren für den Nachrichtenschlüssel EME-OAEP (RSAES-OAEP) bis 2023 umgestellt. Die notwendigen Änderungen und die Migrationsstrategie sind in der „Anlage 16 – Security Schnittstelle" der Gemeinsamen Grundsätze Technik dargestellt. Die bei der Umsetzung aufgeworfenen Fragen werden für die Phase des Interoperabilitätstest zunächst als Best Practice wie folgt beantwortet und nach erfolgreichem Test in die „Anlage 16 – Security Schnittstelle" übernommen:

Sachverhalt 1:

Aus Kapitel 2.1.2

„Dies (gemeint sind die Definitionen zum Signaturalgorithmus) *betrifft die* ***Erstellung und Validierung von Signaturen***

- *von Zertifikaten,*
- *der PKCS#10-Zertifizierungsanfragen,*
- ***der PKCS#7-Zertifififizierungsantworten****,*
- *der eigentlichen Nachrichten im Nachrichtenaustausch"*

Problem: PKCS#7-Zertifikatsantworten sind aber gar nicht signiert, sondern enthalten eine unsignierte Liste von Zertifikaten

Lösung: Die Zeile „der PKCS#7-Zertififizierungsantworten" wird gelöscht, da die Zertifikate in der betroffenen Liste schon aufgeführt werden und die PKCS#7-Zertifizierungsantworten keine anderen Zertifikate als diese enthalten. Die Aufnahme der Zertifikatskette in die PKCS#7-Zertifizierungsantworten ist in Abschnitt 5.9.3.4 definiert.

Sachverhalt 2:

Kapitel 2.1.2.1 sind die RSA-PSS-Parameter wie folgt definiert:

```
RSASSA-PSS-params ::= SEQUENCE {
   hashAlgorithm      [0] HashAlgorithm      DEFAULT sha1,
   maskGenAlgorithm   [1] MaskGenAlgorithm   DEFAULT mgf1SHA1,
   saltLength         [2] INTEGER            DEFAULT 20,
   trailerField       [3] TrailerField       DEFAULT trailerFieldBC
}
```

In 2.1.2.1.4 heißt es dann weiter:

Für das Feld TrailerField ist gemäß RFC-8017 ein fester Wert vorgesehen:

TrailerField ::= INTEGER { trailerFieldBC(1) }

Problem: Da das trailerField einen Default hat, ist es optional. D.h. wenn es fehlt, wird als Inhalt der Default (trailerFieldBC=1) angenommen, der sowieso der einzige erlaubte Wert ist. Nun ist aber nicht klar, ob das mit der Profilierung in der Security-Schnittstelle konform geht das man ein Default einfach leer lässt oder ob die Formulierung in 2.1.2.1.4 („ist ein fester Wert vorgesehen") ein explizites Vorhandensein des trailerField vorsieht. Bei allen anderen Feldern in *RSASSA-PSS-params* ist die Situation klar, weil der Inhalt vom Default-Wert abweicht, so dass diese explizit gesetzt werden müssen.

Lösung: In RFC4055, auf den sich die Security-Schnittstelle bezieht, wird das Weglassen dieses Parameters zwingend vorgeschrieben (https://tools.ietf.org/html/rfc4055#section-3.1):
„Implementations that perform signature generation **MUST omit the trailerField field,** indicating that the default trailer field value was used."

Sachverhalt 3:

In 2.1.4.1 sind die OAEP-Parameter definiert:

RSAES-OAEP-params ::= SEQUENCE {
hashAlgorithm [0] HashAlgorithm DEFAULT sha1,
maskGenAlgorithm [1] MaskGenAlgorithm DEFAULT mgf1SHA1,
pSourceAlgorithm [2] PSourceAlgorithm DEFAULT pSpecifiedEmpty

HashAlgorithm und MaskGenAlgorithm sind zu setzen und soweit klar.

Zu PSourceAlgorithm heißt es unter 2.1.4.1.3:

Als Default-Wert für das Label ist in [RFC-8017] die Angabe eines leeren Labels vorgesehen:

pSpecifiedEmpty PSourceAlgorithm ::= {
algorithm id-pSpecified,
parameters EncodingParameters : emptyString
}

emptyString EncodingParameters ::= "H

Problem: Im Prinzip das gleiche Problem wie Sachverhalt 2. Auch hier stellt sich die Frage, ob der Default-Wert (Leerstring) explizit angegeben werden muss oder auch weggelassen werden kann

Lösung: Analoges Vorgehen zu Sachverhalt 2; der Default-Parameter pSourceFunc gemäß RFC4055 ist wegzulassen.

RFC4055 sagt dazu (https://tools.ietf.org/html/rfc4055#section-4.1):
"Implementations that perform encryption **MUST omit the pSourceFunc field** when a zero length P value is used, indicating that the default value was used."

Sachverhalt 4:

Zu Testzeitpunkt müssen an den Test teilnehmende DAVen mit neuer Software ausgestattet werden. Änderungen der Algorithmen betreffen auch den Zertifizierungsprozess.

Problem: Die Softwareersteller können aber einige Dinge nur „im eigenen Saft" testen, z.B Korrektheit die Zertifizierungsanfrage (Sachverhalt1). Wenn es hier bei den Interoptest zu Problemen kommt, können die eigentlichen Verschlüsselungstests gar nicht durchgeführt werden.

Lösung: Es wird die Möglichkeit geben, im Januar vor den Interoperabilitätstests Zertifikats-Requests mit neuen Algorithmen mit dem Trust Center testen zu können. Das Trust Center wird aber keine detaillierte Fehleranalyse bei abweichenden Zertifikats-Requests liefern. Es können aber positive Zertifikats-Requests als Beispiel genutzt werden, wenn ein SWE noch ein Problem hat.

Sachverhalt 5:

Es geht um eine Binär-Inkompatibilität bei SignedData zwischen dem alten PKCS#7 und dem neuen CMS. Die Spezifikation vermischt hier das Version-Tags (1) des alten PKCS#7 und die geänderte EncapsulatedContent Struktur, die es nur in CMS gibt (Versions 3 oder höher gibt). Mit der Festlegung, dass "version" 1 sein soll, muss bei SignedData die orignal-PKCS#7 v1.5-Definition von SignedData verwendet werden, dort gibt es keine "encapContentInfo".

Problem: Die aktuelle Doku erzeugt einen Widerspruch weil sie das (binär-incompatible) Encoding von CMS für eContentInfo beschreibt, aber mittels der "version = 1" in der SignedData PDU die Verwendung das PKCS#7 v1.7-Encoding für eContentInfo festlegt:

> *3.2.1 Aufbau der Teildatenstruktur Content vom ContentType „SignedData"*
> *Der Typ SignedData hat folgende Syntax:*
>
> *SignedData ::= SEQUENCE {*
> *version CMSVersion,*
> *digestAlgorithms DigestAlgorithmIdentifiers,*
> *encapContentInfo EncapsulatedContentInfo,*

certificates [0] IMPLICIT CertificateSet OPTIONAL,
crls [1] IMPLICIT CertificateRevocationLists OPTIONAL,
signerInfos SignerInfos }

3.2.1.1 Datenfeld „version“
Das Feld version enthält die Versionsnummer der Syntax.

Profilierung:
Für das Datenfeld version soll in dieser Datenstruktur immer der Wert „1“ eingesetzt werden.

Die Verwendung der encapContentInfo von CMS (Version 3+) in Verbindung mit der SignedData.CMSVersion = 1 ist meines Erachtens ein Widerspruch.

3.2.1.3 Datenfeld „encapContentInfo“

EncapsulatedContentInfo ::= SEQUENCE {
eContentType ContentType,
eContent [0] EXPLICIT OCTET STRING OPTIONAL }

Profilierung:
Gemäß [IMMTP3] wird hierfür der eContentType id-data mit der OID 1.2.840.113549.1.7.1 verwendet, die angibt, dass nicht interpretierte binäre Daten in das Teildatenfeld eContent eingetragen wurden.

Bei Verwendung der SignedData.CMSversion = 1 müsste das original PKCS#7-Encoding verwendet werden:

https://tools.ietf.org/html/rfc2315#9.1

9.1 SignedData type

The signed-data content type shall have ASN.1 type SignedData:

SignedData ::= SEQUENCE {
version Version,
digestAlgorithms DigestAlgorithmIdentifiers,
contentInfo ContentInfo,
certificates
[0] IMPLICIT ExtendedCertificatesAndCertificates
OPTIONAL,
crls
[1] IMPLICIT CertificateRevocationLists OPTIONAL,
signerInfos SignerInfos }

7. General syntax

The general syntax for content exchanged between entities according to this document associates a content type with content. The syntax shall have ASN.1 type ContentInfo:

ContentInfo ::= SEQUENCE {
contentType ContentType,
content
[0] EXPLICIT ANY DEFINED BY contentType OPTIONAL }

ContentType ::= OBJECT IDENTIFIER

Lösung: Da in der aktuellen Profilierung weder Zertifikate anderer Typen, noch CRLs (normale oder anderer Typen), noch Attributszertifikate (Version 1 oder 2) oder eine SignerInfo in Version 3 oder ein eContentType ungleich id-data vorhanden ist, ist der bisher verwendete Wert Version=1 völlig korrekt und kollidiert auch nicht mit der Verwendung von encapContentInfo wie sie in CMS definiert ist.

Sachverhalt 6:
Die Security-Schnittstelle definiert in 2.1.2.1.1 den HashAlgorithm, der für PSS zu verwenden ist, und bezieht sich dabei auf RFC8017:

sha256 HashAlgorithm ::= {
algorithm id-sha256,
parameters SHA256Parameters : NULL
}

Eine analoge Definition wird auch in 2.1.4.1.1 für den HashAlgorithm in den OAEP-Parametern vorgenommen, allerdings ohne konkreten Hinweis auf einen RFC.

Problem: Hinsichtlich RFC8017 ist diese Definition durchaus korrekt. Zieht man den – ebenfalls in der Security-Schnittstelle zitierten RFC4055 zurate, merkt man, dass die Frage, ob der HashAlgorithm einen Parameter NULL hat oder nicht, wohl historisch gesehen nicht ganz so eindeutig ist.

RFC4055 sagt dazu:

> There are two possible encodings for the AlgorithmIdentifier parameters field associated with these object identifiers. The two alternatives arise from the loss of the OPTIONAL associated with the algorithm identifier parameters when the 1988 syntax for AlgorithmIdentifier was translated into the 1997 syntax. Later the OPTIONAL was recovered via a defect report, but by then many people thought that algorithm parameters were mandatory. Because of this history some implementations encode parameters as a NULL element while others omit them entirely. **The correct encoding is to omit the parameters field**; however, when RSASSA-PSS and RSAES-OAEP were defined, it was done using the NULL parameters rather than absent parameters.
>
> **All implementations MUST accept both NULL and absent parameters as**

legal and equivalent encodings.

In dieser etwas unübersichtlichen Lage ist jetzt die Frage, wie das im Gesundheitswesen gehandhabt werden soll.

Betroffen sind:

- Zertifikate und Zertifikatsanfragen, die mit RSA-PSS signiert sind
- Daten
    - Die mit RSA-PSS signiert sind
    - Die mit RSA-OAEP verschlüsselt sind

Lösung: Da sich RFC8017 und RFC4055 in ihren Aussage zu diesem Sachverhalt nicht 100%ig eindeutig sind und letzterer verlangt, dass beides durchgelassen wird, wird sowohl das Weglassen des Parameterfeldes als auch der Parameter mit dem Wert „NULL“ zugelassen und die NULL wird als OPTIONAL definiert. In die Security-Schnittstelle wird ein Hinweis aufgenommen, dass sie die o.g. RFCs inhaltlich widersprechen und der RFC 4055 maßgeblich ist.