

OSCI-Bibliothek (.NET) – Versionshistory

Version	Datum	Änderungen gegenüber Vorversion
1.00	24.03.2004	./.
1.01	02.04.2004	ID-Schreibweise in XML-Dokumenten korrigiert; Bei paralleler Mehrfachsignatur mit identischem Zertifikat wird nun jede Signatur geprüft; kleinere Korrekturen vorgenommen
1.02	30.04.2004	Geringe Bugfixes
1.03	23.06.2004	Korrigenda vom 10.06.2004 zur OSCI-Spezifikation 1.2 eingearbeitet, ContentPackageInterface erweitert, Zertifikatsreferenzen vereinheitlicht, Bugfixes, ProcessCard-Operationen erweitert
1.04	08.07.2004	Dialogeröffnung mit Testintermediär nur via InitDialog-Nachricht möglich, FeedbackObject-Klasse eingeführt, die Methode ContentContainer.checkAllSignatures() wirft nun bei Nichtvorhandensein einer Signatur eine Exception, MediateDelivery erlaubt nun auch Messageld ohne Subject (laut Spec), Default bei QualityOfTimeStamp-Eigenschaften ist nun „plain“
1.1	05.01.2005	Diverse Bugfixes, z.B. ein Problem mit Attachments, die in verschachtelten ContentContainern referenziert werden, beseitigt. Laufzettelaufträge für mehrere Message-Ids erweitert. Diverse Anpassungen und Änderungen mit dem Ziel der Kompatibilität mit Apache-XML Implementierungen. Base64-Codierung der Attachments als Transfer-Encoding. Transport-Interface um Methode „newInstance()“ erweitert (Threadsicherheit in „automatischen“ Clients).
1.1.1	29.06.2005	Bugfix: innere ContentContainer werden nun auch geschrieben. Verfeinerung: ProcessCardBundle.writeToStream(...) schreibt jetzt Messagelds Base64-codiert. Darüber hinaus wurden diverse Sysouts entfernt.
1.2	03.02.2006	Das Schließen von http-Connections wurde verbessert. Es wurden zahlreiche Änderungen mit dem Ziel der Kompatibilität zu kommerziellen XML-Implementierungen vorgenommen. Hervorzuheben ist hier die Entfernung von Leerzeilen in dem Attachment MIME-Container. Es wurden Änderungen, um die Namespace-Präfixe variabel zu gestalten, vorgenommen. Grenzen in der Realisierung haben sich durch XML-Signature ergeben, da ansonsten Signaturen unverschlüsselter Inhaltsdaten bei Änderungen der Präfixe ungültig würden. Die Base64-Codierung für den Transportumschlag und Attachments ist nun optional, bleibt jedoch Default. Diese Default-Belegung sollte nur bedacht geändert werden, da sich hieraus Inkompatibilitäten bei der Kommunikation mit vorherigen Versionen der OSCI-Bibliothek (und Intermediären) ergeben. Im DialogHandler können nun mehrere Default-Supplier, also mehrere Privatschlüssel, für die Entschlüsselung eingehender Nachrichten (und ggf. Signatur der Antworten) gesetzt werden. Die zusätzlichen Methoden hasCipherCertificate() und hasSignatureCertificate() sind nun in der Role-Klasse verfügbar. Es wurde ein Fehler im Rahmen FetchProcessCard bei SelectionRule SELECT_BY_MESSAGE_ID behoben.

		Für die Log-Klasse kann nun das Verhalten über Umgebungsvariablen gesteuert werden (Loglevel, Logausgabe in Datei, s. Doku)
1.2.1	24.02.2006	Es wurde eine zweite Version für das .NET-2.0-Framework hinzugefügt, die nicht mehr die capicom.dll und die CertGetKey.dll benötigt. Außerdem wurde der Quellcode bereinigt und zwei Probleme der Version 1.2 beseitigt (Reihenfolge der ContentContainer-/EncryptedData –Tags, unsignierte Nachrichten)
1.2.2	14.06.2006	Problem mit signierten verschachtelten ContentContainern, die Attachments enthalten, behoben. Verlust des Base64-Transformers der Inhaltsdatensignatur beim Laden gespeicherter Nachrichten beseitigt. Neue Methode zum Laden gespeicherter Nachrichten mit Prüfung der Nachrichtensignatur in der Klasse StoredMessage. Zwei Fehler bei der Serialisierung empfangener Nachrichten behoben.
1.2.3	06.12.2006	Kleinere Probleme beseitigt.
1.2.4	28.02.2007	AES-Unterstützung aktiviert (vgl. Patch 1.2.3.01). Problem mit langen Seriennummer von Zertifikaten im .NET 2.0-Framework behoben. Fehlende Bedienung des OnlineChecked-Flags beim Parsen von Laufzetteln korrigiert. Unterstützung zusätzlicher MIME-header in Attachments. Lock-statements in Log-Klasse für Threadsicherheit beim Loggen in Dateien.
1.2.5	17.4.2007	Kleinere Änderungen.
1.2.6	05.09.2007	Begrenzung der Attachmentgröße auf 2 GB beseitigt. Kleinere Änderungen.
1.3	10.7.2008	Erweiterung auf neue Hashalgorithmen gemäß Korrigenda der OSCI 1.2 Transport Spezifikation v. 10.4.2008. Hierzu Erweiterung des Signer-Interfaces. Übergangsweise wird SHA-1 weiter unterstützt. Für die Unterstützung der neuen Hashalgorithmen ist die Installation eines Patches für das .NET Framework erforderlich (s. Datei „index.html“ im Verzeichnis „OSCI-Bibliothek-.NET“). Die Verwendung der Quelldateien für mehrsprachige Ausgaben wurde vereinfacht (s. Klasse ResourceBundle) Die Unterstützung des .NET Frameworks 1.1 wurde eingestellt.
1.3.1	15.10.2008	Installation vereinfacht (Einbindung des Microsoft-Crypto-Patches als normale DLLs, Sprachdateien im Verzeichnis der osci-bib-DLL, deutsche Standard-sprachdatei in DLL kompiliert). Zur Kontrolle der Signaturstärke empfangener Signaturen zwei neue Methoden OSCIMessage.hasWeakSignature(DateTimeOffset) und ContentContainer.hasWeakSignature(Role, DateTimeOffset). Neues Attribut „OSCIMessage“ in OSCICollectionException. Problem mit ungültigen Nachrichtensignaturen bei seriell verschlüsselten Inhaltsdaten behoben. Default-Algorithmus für symmetrische Verschlüsselung auf AES-256 umgestellt.
1.3.2	03.06.2010	Anpassung an Ablauf des Hashalgorithmus „RIPEMD-160“ zum 31.12.2010. Die Prüfmethode hasWeakSignature(...) liefern bei Prüfung ohne Datumsangabe für diesen Algorithmus „true“. Methoden für frei formulierte feedback-Texte in synchronen Szenarien auf Empfängerseite. Kleinere Überarbeitungen.
1.4	10.11.2010	Vorbereitung für die Aufnahme von Signaturzeitpunkten in Inhaltsdatensignaturen (rudimentäre XAdES-Unterstützung). Diverse kleinere Überarbeitungen.
1.5	19.10.2011	Unterstützung elliptischer Kurven als Signaturalgorithmen (Korrigenda v. Oktober 2011). Umstellung auf Bouncy Castle 1.7 als Crypto-Provider,

		Microsoft-Crypto-Patch entfernt. Kleinere Überarbeitungen.
1.6	16.04.2014	<p>Unterstützung von OAEP-Algorithmen für Verschlüsselung und Signaturen gemäß Korrigenda der OSCI 1.2 Transport Spezifikation v. 20.2.2014.</p> <p>Implementierungen der Schnittstelle „de.osci.osci12.extinterfaces.crypto.Decrypter“ müssen für die Verwendung von RSAES-OAEP-ENCRYPT um eine entsprechende Methode erweitert werden.</p> <p>Auslieferung der Libraries in zwei Versionen für die .NET-Frameworks 3.5 und 4.0 (letzteres auch unter Framework 4.5 getestet).</p>
1.7	08.03.2017	<p>Es wird unterbunden, dass für mehrere Inhaltsdaten gleiche Referenz-IDs verwendet werden oder nachträglich dahingehend geändert werden können.</p> <p>Einbindung der SHA3-Hash-Algorithmen</p> <p>Aktualisierung von Bouncy Castle</p> <p>API-Änderungen:</p> <p>Anpassung der Namenskonventionen (Namespaces, Methoden) an .NET</p> <p>Sicherheits-Update:</p> <p>Anpassungen an den Prüfungen von XML-Dokumenten und Aufrufen des verwendeten XML-Parsers.</p> <p>Die Prüfung auf eindeutige IDs innerhalb verschlüsselter Daten (EncryptedDataOSCI) ist in der API konfigurierbar (Osci.GlobalSettings; standardmäßig aktiviert).</p> <p>Für die Verwendung des CBC-Modus sind serverseitig zusätzliche Sicherheitsmaßnahmen nötig. Daher wurde ein neuer Modus (GCM) hinzugefügt, der langfristig den CBC-Modus ablösen wird.</p>
1.8	30.01.2018	<p>Es wurden verschiedene Änderungen vorgenommen, um die paketierte Übertragung von Nachrichten, die in der Erweiterung der OSCI-Spezifikation bzgl. effizienter Datenübertragung beschrieben ist, zu ermöglichen.</p> <p>Es gibt Methoden um die zusätzlichen Header, wie die <code>FeatureDescription</code> und die <code>ChunkInformation</code>, zu erstellen. Außerdem gibt es Methoden um die neuen Nachrichtentypen <code>PartialStoreDelivery</code> und <code>PartialFetchDelivery</code> aufzubauen sowie die Response Nachrichten zu verarbeiten. Die Aufteilung einer großen Nachricht in Pakete wird an einem Beispiel demonstriert.</p>
1.8.2	05.11.2018	<p>Umstellung auf die neuere Version des BouncyCastle-Security-Providers 1.8.4</p> <p>Sicherheits-Update:</p> <p>Es wurden Verbesserungen im MIME-Parser und bei der Strukturprüfung der Signaturen vorgenommen.</p> <p>Bei der Verwendung des CBC-Modus wird im Client-Dialog automatisch auf GCM gewechselt, wenn Client und Server diesen Modus unterstützen.</p>
1.8.3	12.11.2018	<p>Sicherheits-Update:</p> <p>Zusätzliche Verbesserungen bei der Strukturprüfung der Signaturen.</p>
1.9.0 1.9.0-internal-1	27.01.2020	<p>Verwendung von AES-256-GCM anstelle von AES-256-CBC als Standardmodus für die symmetrische Verschlüsselung</p> <p>Verwendung von RSA-OAEP anstelle von RSAES-PKCS1-v1_5 als Standardmodus für die asymmetrische Verschlüsselung</p> <p>Der Initialisierungsvektor bei Verwendung von AES-GCM lässt sich anpassen: 12 Byte alias 96 Bit, empfohlen gemäß XML-Encryption-Standard, oder 16 Byte alias 128 Bit, abwärtskompatibler alter Standardwert</p>

		<p>Der asymmetrische Verschlüsselungsalgorithmus wird innerhalb eines fortlaufenden OSCI-Dialogs korrekt geparkt</p> <p>Beim der Verarbeitung von großen Nachrichten (> ca. 50 MB) wird eine OutOfMemoryException vermieden</p> <p>Umstellung auf die neuere Version des BouncyCastle-Security-Providers 1.85</p>
2.0.1	05.02.2021	<p>Umstellung des Standardwerts für die Länge des Initialisierungsvektors bei der Verwendung von AES-GCM von 16 Byte bzw. 128 Bit auf den empfohlenen Wert 12 Byte bzw. 96 Bit</p> <p>Umstellung auf die neuere Version des BouncyCastle-Security-Providers 1.89</p> <p>Element-Reihenfolge im Nachrichtentyp „ResponseToAcceptDelivery“ entspricht jetzt vollständig der Spezifikation</p> <p>Kleinere Optimierungen des Codes zur Verbesserung der Qualität</p>