Basismodul VAU-Kanal, Version 1.3.7

Überblick

Das vorliegende Modul zeigt exemplarisch, wie das im Spezifikationsdokument [gemSpec_Krypt] normativ festgelegte Kommunikationsprotokoll zwischen VAU und ePA-Clients implementiert werden kann.

Spezifikationsbasis ist "Release 4.0.2".

Aufbau des Moduls

Die Implementierung erfolgt in Java (Version 11). Buildsystem ist Apache Maven (Version 3.6.1).

- **vauchannel** ist das maven reactor module, das die einzelnen Teilmodule in einem Build baut.
- **vauchannel-contract-2-java** legt das Format der ausgetauschten JSON-Nachrichten per JSON Schema (draft-04) und damit unabhängig von der Implementierungssprache fest. Aus den Schemas werden Java Klassen generiert.
- **vauchannel-protocol** implementiert das Protokoll mit der Verarbeitung der Nachrichten in der Javaklasse `VAUProtocol.java`. Die Definition des Kommunikationsprotokoll zwischen VAU und ePA-Clients aus [gemSpec_Krypt] Kapitel 6 werden in `VAUProtocol.java` detailliert kommentiert auf die Java-Implementierung gemappt. Abgedeckt sind
  - der Handshake,
  - der verschlüsselte Nutzdatentransport,
  - das Fehlerhandling. Alle benötigten kryptografischen Funktionen werden über ein Interface `VAUProtocolCrypto` angesprochen. Die konkrete Implementierung des Interfaces ist damit unabhängig von der Protokollimplementierung. Sessioninformationen werden in der `VAUProtocolSession` abgelegt.
- **vauchannel-cxf** zeigt die Einbettung in die WebService-Bibliothek Apache CXF (XML/SOAP für die fachlichen Services und JSON/REST für das VAUProtokoll). Der VAU-Kanal startet und endet in dieser Einbettung in CXF-spezifischen Interceptoren. Die konkrete Implementierung des VAUProtokolls mit Kryptofunktionen und Ablage der Sessioninformationen wird als "dependency injected".
- **vauchannel-server** zeigt exemplarisch die Umsetzung der serverseitigen Endpunkte für Handshake und fachlichen Service (hier ein sayHello-Service) als Spring-Boot-Anwendung. Nicht thematisiert:
  - Das Ansprechen von Handshake-Service und OpenContext-Service unter einem Interface, das beispielsweise durch Trennung der Anfragen über den HTTP-Header `Content-Type` in einem vorgelagerten Gateway erfolgen kann.

    - Die Verwaltung mehrerer `VAUProtocolSession`.
- **vauchannel-client** implementiert einen Spring-Boot-Client, der gegen einen laufenden vauchannel-server Integrationstests ausführt.

Bauen

Das Modul **vauchannel** wird samt seinen Untermodulen wie folgt gebaut:

```
mvn clean install
```

Das Ergebnis des Builds sollte etwa so aussehen:

```
[INFO] ------------------------------------------------------------
[INFO] Reactor Summary for vauchannel 1.3.6:
[INFO]
[INFO] vauchannel ............................ SUCCESS [  0.256 s]
[INFO] vauchannel-contract-2-java ............. SUCCESS [  2.333 s]
[INFO] vauchannel-protocol .................... SUCCESS [  6.351 s]
[INFO] vauchannel-cxf ......................... SUCCESS [  0.686 s]
[INFO] vauchannel-server ...................... SUCCESS [  7.882 s]
[INFO] vauchannel-client ...................... SUCCESS [  1.019 s]
[INFO] ------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------
```

Starten

Den Server starten

```
cd vauchannel-server/target
java -jar vauchannel-server-1.3.7.jar
```

und vom Client aus Integrationstests ausführen

```
cd vauchannel-client
mvn verify -Ptest
```

Release-Notes

V1.3.7, 2021.03.12

Die Generierung der OCSPResponse wurde gemäß Anforderung GS-A_4693-01 [gemSpec_PKI] um die OCSP-Extension "certHash" erweitert.
Außerdem wird nun in der OCSP-Response gemäß RFC6960 folgendes sichergestellt:

- Das Feld CertID wird mit den Informationen des CA-Zertifikates befüllt
- Die ResponderID wird basierend auf dem Key des OCSP-Signers berechnet
- Das öffentliche Zertifikat des OCSP-Signers wird in der Response mitgeliefert

V1.3.6, 2020.10.28

- Die Unterstützung der Deserialisierung des "Time"-Felds in VAUServerError-Nachrichten gemäß der Kodierung nach ISO-8601 (Anforderung "A_16851 - VAU-Protokoll: VAUServerError-Nachrichten") wurde optimiert.

V1.3.5, 2020.08.06

- Fehlermeldungen, die vom Kontextmanager zurückgegeben werden, werden nicht als VAUServerError transportiert.
- Eine leere OCSPResponse wird in der Nachricht VAUClientSigFin gemäß A_17070-01 als leere Zeichenkette übertragen.
- Bei der AES-Verschlüsselung im VAU-Kanal muss gemäß A_16945-01 (Client) und A_16952-01 (Server) der Initialisierungsvektor den Nachrichtenzähler enthalten. Dieser wurde bisher nicht in den Initialisierungsvektor übernommen, was durch die Änderung korrigiert wird. Da der Wert jeweils nicht von der Gegenseite geprüft wird, hat die Korrektur keinen Einfluß auf die Interoperabilität.

V1.3.4, 2020.07.27

Bei der Prüfung der Clientidentität an Hand der Nachricht VAUClientSigFinMessage wurden die Hash-Werte zu VAUClientHelloData und VAUServerHelloData aus der Nachricht VAUClientSigFinMessage verwendet. Das wurde korrigiert: Bei der Prüfung werden jetzt die vom Server verifizierten Hashes verwendet.

V1.3.3, 2020.05.07

Das Encoding aus dem HTTP-ContentType-Header wird im CXF-AESInterceptor nun korrekt durch den verschlüsselten Kanal geleitet. Dadurch wird der HTTP-Content-Type für Typ (2) Nachrichten spezifikationskonform gemäß "A_16884 - VAU-Protokoll: Nachrichtentypen und HTTP-Content-Type".

V1.3.2, 2020.03.26

Die mit V1.3.1 eingeführte Hülle um die ECDSA-Signatur wurde wieder herausgenommen. Der Stand diesbezüglich entspricht jetzt wieder dem von V1.3.0. Hintergrund: Aus Gründen der Kompatibilität zwischen verschiedenen VAU-Protokollversionen sollen Parameter, welche die ECDSA/RSA-Signatur charakterisieren, zukünftig mit der Signatur übertragen werden. Anstatt aber auf ASN.1-Ebene diese Parameter zu platzieren ist angedacht, in einer zukünftigen Spezifikationsversion die Parameter auf JSON-Ebene zu transportieren.

V1.3.1, 2020.03.19

- **Neu:** Serverzertifikat wird gegen TSL sowie per OCSP geprüft.
- ~~**Fehlerbehebung:** Die Einbettung der ECDSA-Signaturen wird in Aktor korrigiert und an die Spezifikation angepasst. Zur Anforderung [A_16901-0] ist erläutert „Die ECDSA-Signatur MUSS nach [TR-03111#5.2.2. X9.62 Format] (inkl. OID "ecdsa-with-Sha256") kodiert sein." Die Sequenz mit den zwei „Integer" (r und s aus der ECDSA-Signatur) ist umhüllt von einer Sequence und am Anfang ist die OID „ecdsaWithSHA256" aufgeführt. Diese Hülle hatte bisher bei allen VAU-Kanal ECDSA-Signaturen im Aktor gefehlt. Wie die Hülle aussieht erläutert das Beispiel in gemSpec_Krypt#5.2 nach „Und am Ende des Zertifikats befindet sich die ECDSA-Signatur":~~

```
535  10:   SEQUENCE {
537   8:     OBJECT IDENTIFIER ecdsaWithSHA256 (1 2 840 10045 4 3 2)
       :     }
547  73:   BIT STRING, encapsulates {
550  70:     SEQUENCE {
552  33:       INTEGER
       :         00 A5 0E AA 18 74 F1 F7 B2 2C 77 38 28 58 7F 7F
       :         5B 7D B2 B7 8A E4 B9 D5 22 B3 4C 71 19 9E 3C 60
       :         E5
587  33:       INTEGER
       :         00 93 43 8A 6E A2 5E 58 60 80 19 62 4E F8 99 F8
       :         9D DC 90 4E BC C0 55 FD 78 0F 57 65 A9 4B FF 7D
       :         CA
       :       }
       :     }
       :   }
```</del>

V1.3.0, 2020.03.11
* **Neu:** Anpassung auf Spezifikationsstand zum Online-Produktivbetrieb, Release 3.1.3.

V1.2.0, 2019.12.18
* **Optimierung:** Vollständiger HTTP-Body der entschlüsselten Nachricht wird geloggt.
* **Optimierung:** Bessere Unterstützung für die Umsetzung der Anforderungen
"A_14545 - Komponente ePA-Dokumentenverwaltung - Operationen des Dokumentenmanagements nur über s
und
"A_18714 - Komponente ePA-Dokumentenverwaltung - Verhalten des Kontextmanagements bei ungeöffnete
* **Optimierung:** Unit-Tests zum Fehlerhandling erweitert

V1.1.1, 2019.10.23
* **Fehlerbehebung:** Eine Einschränkung für den Transport großer Dokumente über den VAU-Kanal im

V1.1.0, 2019.09.18
* **Neu:** Anpassung auf Spezifikationsstand zum Online-Produktivbetrieb, Release 3.1.2.
* **Neu:** Insbesondere wurde Change C_7029 implementiert, der für einen verschlüsselten Transpor
des Content-Type HTTP-Headers sorgt.

V1.0.0, 2019.09.03
* **Neu:** Initiale Implementierung gemäß der Spezifikationsdokumente zum Online-Produktivbetrie