IMPLEMENTATIEHANDLEIDING MITZ

Security tokens generiek

Versie: 3.8.0
Status: Definitief
Datum: 21 juni 2021

WIJZIGINGENBEHEER

Versie	Hoofdstuk	Auteur	Opmerkingen
3.6.0.0		DD	Concept versie Security Tokens Generiek
3.6.0.0		FS	Ter goedkeuring versie 3.6.0.0 release na interne review
3.6.0.0		AV	Definitieve versie 3.6.0.0
3.7.0.TV		FS	Uitwerking migratie v3.7.0 interne review
3.7.0.TG		FS	Uitwerking migratie v3.7.0 externe review
3.7.0		AV	Definitieve versie 3.7.0
3.8.0.TV		FS	Uitwerking toestemmingsknop v3.8.0 interne review
3.8.0.TG		FS	Uitwerking toestemmingsknop v3.8.0 externe review
3.7.0		AV	Definitieve versie 3.8.0

DOEL

Dit document legt de basis voor de verschillende beveiligingsimplementatiehandleidingen waarin security tokens voor een specifieke toepassing concreet beschreven worden. Dit document beschrijft wat security tokens zijn, wat het doel is van security tokens en hoe security tokens in de Mitz-architectuur worden ingezet.

Dit document is bedoeld voor softwareontwikkelaars van XIS'en. De lezer wordt verondersteld kennis te hebben van [XML], [Namespaces], [SOAP], [HTTP], [WSDL] en [IH Transport].

INHOUDSOPGAVE

1 INTRODUCTIE

Het programma “Online Toestemmingsvoorziening (OTV)” streeft naar het gebruik van open standaarden, omdat daardoor een open koppelvlak ontstaat waarop de leveranciers die dat wensen hun producten en diensten kunnen koppelen. Daarnaast zijn open standaarden van belang omdat componenten die al gebouwd en gebaseerd zijn op deze standaarden, hergebruikt kunnen worden en dat is efficiënter en goedkoper dan maatwerk ontwikkelen en bouwen. In die situatie zouden bijvoorbeeld bestaande internationale componenten ‘slechts’ geconfigureerd hoeven te worden voor de Nederlandse situatie, in plaats van volledig opnieuw geprogrammeerd.

Dit document beschrijft wat security tokens zijn, wat het doel is van security tokens en hoe security tokens in de Mitz-architectuur worden ingezet.

2 KADERS EN UITGANGSPUNTEN

2.1 EXTERNE NORMEN EN KADERS

Op het gebied van beveiliging in de zorg zijn verschillende normen en wetten van kracht, zowel op Europees als Nederlands niveau. De NEN 7510 norm is de Nederlandse Norm voor Informatiebeveiliging in de Zorg en is van toepassing op alle zorgaanbieders.

Bij de uitwisseling van informatie in de zorg speelt de wetgeving een belangrijke rol. Een uitgebreide behandeling hiervan valt buiten de reikwijdte van dit document, echter de wet Wbsn-z wordt hier kort benoemd.

Wet Gebruik Burgerservicenummer in de Zorg (Wbsn-z)

De Wbsn-z (sinds 2008) regelt dat in alle berichtgevingen tussen zorgaanbieders onderling het *burgerservicenummer* (BSN) aanwezig **moet** zijn om persoonsverwisseling en daardoor medische fouten te voorkomen. Een burgerservicenummer mag door de zorgaanbieder worden geregistreerd nadat de betrokkenen zich heeft gelegitimeerd met een wettelijk identificatiedocument.

2.2 NAMESPACES

Wanneer in dit document in voorbeeld fragmenten niet gedeclareerde namespace prefixes worden gebruikt, worden de volgende namespaces bedoeld.

Prefix	Namespace URI
ds	`http://www.w3.org/2000/09/xmldsig#`
soap	`http://schemas.xmlsoap.org/soap/envelope/`
wsu	`http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd`
wss	`"http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"`
saml	`urn:oasis:names:tc:SAML:2.0:assertion`

In XML mag een prefix veranderd worden (mits consequent toegepast). Na ondertekening kan dit echter niet meer, dit zou de handtekening ongeldig maken.

3 HET SECURITY TOKEN

Dit hoofdstuk is niet normatief.

Berichten die worden uitgewisseld op het koppelvlak Mitz-US kunnen worden uitgerust met ondertekende security tokens om de integriteit, authenticiteit en onweerlegbaarheid ervan te waarborgen.

Een security token is opgebouwd uit de volgende componenten:

- verzameling authentieke kenmerken met de daarbij behorende waarden,
- een digitale handtekening,
- randvoorwaarden zoals transformaties, matchingregels, certificaten, etc.

Een security token is niet op te stellen zonder de privé sleutel waarmee het security token wordt ondertekend.

Een security token kan op verschillende manieren worden gebruikt:

- Het security token is aan één of door één entiteit uitgegeven (persoon, machine, etc.) en dient voor bepaalde (authenticatie, autorisatie of logging) doeleinden. Het security token bevat informatie over de afzender en bevat bepaalde kenmerken over een gegeven onderwerp.
- Het security token geeft het recht (autorisatie) om een bepaalde activiteit uit te voeren. Het security token wordt gekoppeld aan interacties.

3.1 OPBOUW

Het security token bevat een aantal basiskenmerken en beveiligingsgerelateerde gegevens. Het security token inclusief digitale handtekening mag dan zonder de privé sleutel niet na te maken zijn, het is wel te kopiëren. Gekopieerde tokens kunnen opnieuw gebruikt of misbruikt worden voor andere acties. De volgende basiskenmerken en beveiligingsgerelateerde gegevens horen bij het security token:

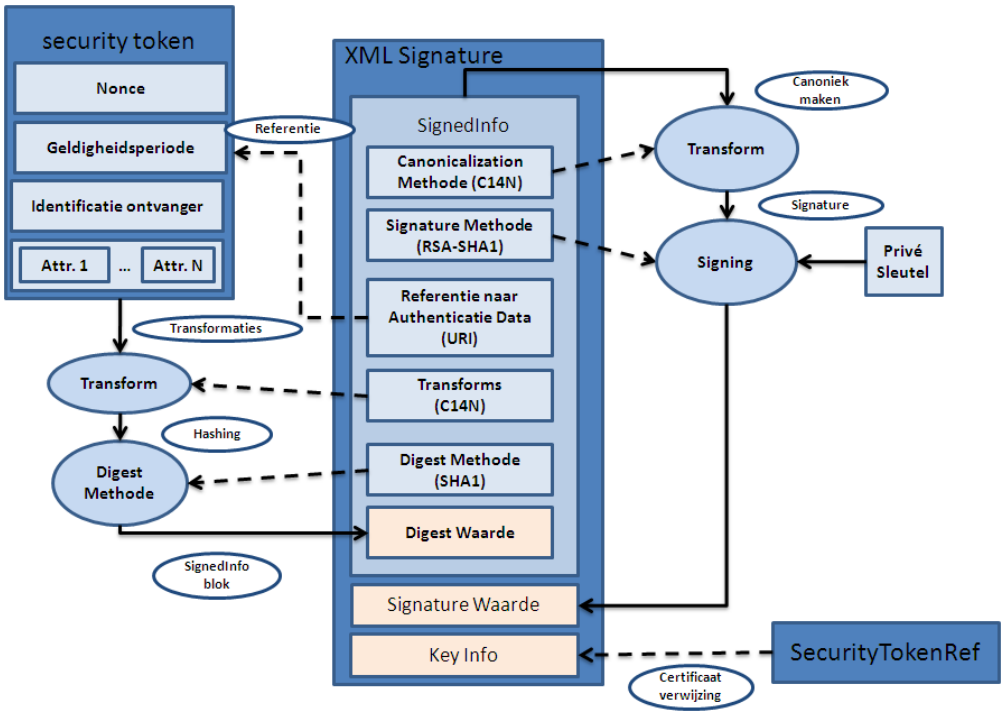
- Uniekheid. Per toepassing moet aangegeven worden of het security token *mondiaal uniek* identificeerbaar moet zijn. Dit kan worden gerealiseerd door een nonce (**N**umber used **ONCE**) in het bericht te stoppen.
- Onderwerp. Het security token heeft een onderwerp waarover de uitgewisselde (beveiligingsgerelateerde) gegevens gaan. Het onderwerp wordt uniek geïdentificeerd en gebruikt bij dialogen tussen partijen (bijvoorbeeld: authenticiteit van een bericht vaststellen op basis van een bericht-id).
- Geldigheid. Elk security token voldoet aan bepaalde condities en geldigheidsvoorwaarden.
- Afzender. De afzender of leverende partij van het security token. Het token bevat een unieke identificatie van de afzender.
- Ontvanger. Afnemer of gebruiker van het security token. Het token bevat een unieke identificatie van de ontvanger. Het doel hiervan is dat de ontvanger de uniciteit van de nonce moet kunnen controleren en de mogelijkheid van hergebruik van een security token bij derden moet worden voorkomen. Echter bij ongeadresseerde verzendingen waarbij het security token wordt gebruikt, is de ontvanger niet bekend.
- Authenticatie. Geeft aan binnen welke authenticatie-context het security token gebruikt wordt als referentiekader voor communicatie tussen de afzender en de ontvanger van het token. De authenticatie-context kan informatie bevatten over de actuele authenticatie methode (bijvoorbeeld de "sterkte" van het gebruikte authenticatiemiddel).
- Middel waarmee het onderwerp in het security token is geauthenticeerd (SAML definieert de details van meer dan 20 verschillende authenticatie methodes).
- Attributen. Aanvullende (essentiële) attributen die een sterkere relatie leggen tussen het bericht en het security token.

Afhankelijk van het type security token, worden bovenstaande componenten in het security token impliciet dan wel expliciet vastgelegd. De componenten worden als aparte paragrafen per beveiligingsimplementatie handleiding verder uitgewerkt.

4 DIGITAAL TEKENEN

4.1 INLEIDING

Om de integriteit te waarborgen wordt (een subset van) het security token ondertekend met behulp van XML Signature. De ontvanger kan dan onomstotelijk vaststellen dat het security token ondertekend is met de privé sleutel behorend bij het certificaat van de verzender.



FIGUUR 1: SECURITY TOKEN MET (SHA1) XML SIGNATURE

4.2 INHOUD VAN DE XML SIGNATURE

De XML Signature bij het security token ziet er als volgt uit (de lange Base 64 strings zijn fictieve waarden en afgekort):

```
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <ds:SignedInfo>
   <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
   <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha256"/>
   <ds:Reference URI="#security_token_Nonce">
     <ds:Transforms>
      <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
      <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
     </ds:Transforms>
     <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
     <ds:DigestValue>xx5...Gus2g=</ds:DigestValue>
   </ds:Reference>
  </ds:SignedInfo>
  <ds:SignatureValue>bULmRGKBlhyPbumKDxrrT...SiGWaarde</ds:SignatureValue>
  <ds:KeyInfo>
   <!-- KEYINFO bevat informatie over gebruikte certificaat -->
  </ds:KeyInfo>
</ds:Signature>
```

Er volgt nu een bespreking van alle XML Signature elementen.

4.2.1 HET <SIGNATURE> ELEMENT

```
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
```

De XML Signature begint met het Signature blok en een verwijzing naar de XML Signature namespace.

Het Signature blok kan onderdeel zijn van het security token of refereert naar het security token.

4.2.2 HET <SIGNEDINFO> ELEMENT

```
<ds:SignedInfo>
```

Vervolgens komt het SignedInfo blok. Dit blok bevat de gegevens die daadwerkelijk getekend worden.

4.2.3 HET <CANONICALIZATIONMETHOD> ELEMENT

```
<ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
```

Het SignedInfo blok zelf moet in de juiste canonieke vorm gebracht worden. Voor de digitale handtekening wordt het Exclusive XML Canonicalization zonder comments gebruikt [EXCC14N]. Deze vorm van canoniek maken, gaat goed om met namespaces van XML documenten die ingebed zijn in andere documenten.

Aangezien het security token wordt samengevoegd met het onderliggend bericht, dat weer is ingebed in een SOAP envelop[1], is het gebruik van Exclusive XML Canonicalization verplicht.

4.2.4 HET <SIGNATUREMETHOD> ELEMENT

```
<ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha256"/>
```

De methode van tekenen. Voor de digitale handtekening wordt gebruik gemaakt van een RSA handtekening over een digest. Voor de gebruikte algoritmes zie 4.3.

4.2.5 HET <REFERENCE> ELEMENT

<ds:**Reference** URI="#security_token_Nonce">

Een referentie naar het token dat getekend is. De referentie is een placeholder en de waarde die wordt overgenomen in een relatieve URI (dus voorzien van een "#" als prefix) moet globaal uniek zijn (i.v.m. het samenvoegen van mogelijk meerdere security tokens in een enkel document).

4.2.6 HET <TRANSFORMS> ELEMENT

```
<ds:Transforms>
  <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
</ds:Transforms>
```

Er worden een aantal transforms gedaan over het token. De minimale set van transforms is de exclusieve canonicalisatie.

[1] In FHIR berichten wordt het security token in de HTTP header geplaatst.

4.2.7 HET <DIGESTMETHOD> ELEMENT

```
<ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
```

Van het canonieke SignedInfo blok van het token wordt een digest berekend. Het gebruik van SHA256 DigestMethod wordt voor eerste implementatie voorgeschreven.

4.2.8 HET <DIGESTVALUE> ELEMENT

```
<ds:DigestValue>xx5...us2g=</ds:DigestValue>
```

De digest over de getekende gegevens wordt opgeslagen, gecodeerd met Base 64. Base 64 is een methode om hexadecimale gegevens, zoals een digest of signature, om te zetten in een teken dat bestaat uit een beperkte set leesbare tekens. Dat is essentieel voor het opnemen in XML, omdat niet ieder willekeurig octet zomaar in XML voor mag komen. Met Base 64 codering is dit geen probleem. Base 64 is een eenduidige code, eenvoudig om te zetten in de oorspronkelijke binaire vorm. Base64 moet gebruikt worden zoals gespecificeerd in [MIME].

```
  </ds:Reference>
</ds:SignedInfo>
```

Einde van de Reference en het SignedInfo blok.

4.2.9 HET <SIGNATUREVALUE> ELEMENT

```
<ds:SignatureValue>bULmRGKBlhyPbumKDxrrT...SiGWaarde</ds:SignatureValue>
```

Over (de canonieke vorm van) het SignedInfo blok wordt een digest berekend waarover een RSA handtekening gezet wordt. Dit volgens de methode zoals vermeld in de eerder vermelde SignatureMethod. De waarde van de RSA handtekening wordt met Base 64 gecodeerd en opgenomen in de XML Signature. De handtekening is dus niet over het token zelf, maar over een digest van een SignedInfo blok, waarin een digest van het token staat.

4.2.10 HET <KEYINFO> ELEMENT

```
<ds:KeyInfo>

  <!-- KEYINFO bevat informatie over gebruikte certificaat -->
</ds:KeyInfo>
```

Er worden gegevens van het gebruikte certificaat opgenomen. Dit wordt in paragraaf 4.4 Certificaten nader toegelicht.

```
</ds:Signature>
```

Einde van de XML Signature.

4.3 ALGORITMES

Voor ieder security token bestaat er een aparte beveiligingsimplementatie handleiding in Mitz. Een dergelijke handleiding specificeert welke algoritmes voor dat specifieke token toegestaan zijn.

Voor het berekenen van hashwaarden wordt SHA-256 gebruikt. Voor de digitale handtekening wordt RSA (asymmetrische encryptie algoritme) gebruikt.

In de XML Signature wordt tweemaal gerefereerd aan deze waarden. Eenmaal voor het `<SignatureMethod>` element en eenmaal voor `<DigestMethod>` element. In deze elementen wordt een subset gebruikt van de URI's zoals gespecificeerd in [XMLSECURI]:

Functie	**Algoritme**	**URI**
Signature	RSA+SHA-256	`<SignatureMethod Algorithm= "http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>`
Digest	SHA-256	`<DigestMethod Algorithm= "http://www.w3.org/2001/04/xmlenc#sha256"/>`

Tabel Mitz.STK.t3100 – Algoritmes

4.4 CERTIFICATEN

Voor ieder security token bestaat er een aparte beveiligingsimplementatie handleiding in Mitz. Een dergelijke handleiding specificeert welke certificaten zijn toegestaan en hoe deze opgenomen worden in het bericht.

Een certificaat koppelt de identiteit van een actor aan een publieke sleutel. Alleen de houder van het certificaat beschikt over de privé sleutel en kent de toegangscode tot het authenticatiemiddel.

Een vertrouwde partij (*trusted third party*) geeft certificaten uit. Deze partij wordt de uitgever (*issuer*) genoemd. Om aan te tonen dat deze uitgever daadwerkelijk het certificaat heeft uitgegeven, ondertekent de uitgever het certificaat met zijn privé sleutel. Controle van deze ondertekening kan plaatsvinden door deze te controleren tegen de publieke sleutel in het certificaat van de uitgever. De controle moet voor de gehele keten een keer plaatsvinden tot aan het vertrouwde (root) certificaat. Daarna kan dit (nu ook vertrouwde) certificaat opgenomen worden in de softwareomgeving.

Twee CA's (*Certificate Authority*) spelen een belangrijke rol in Mitz:

- PKIoverheid. PKIoverheid heeft eigen CA's. Deze vallen allen onder de Root CA van de Staat der Nederlanden. Daaronder is een opdeling in verschillende domeinen, onder andere één voor burgers en één voor overheid en bedrijven (organisaties). De huidige hiërarchie (G2) worden certificaten ondertekend met SHA-256 en worden langere sleutels gebruikt. Deze hiërarchie wordt de tweede generatie (G2[2]) genoemd. PKIoverheid is in 2017 over gegaan op een nieuwe hiërarchie, de derde generatie G3. Zie informatie op de Logius website.
- UZI-register van het CIBG. Het UZI-register is een intermediate CA onder PKIoverheid. Het CIBG is een uitvoeringsorganisatie van het ministerie van VWS, welke een programma voert met diverse CA's voor de zorgsector. Zij voeren dit uit onder de PKIoverheid richtlijnen en hiërarchie. Het CIBG treedt hierbij zowel als Register Authority als Certificate Authority op. Er worden certificaten (voor zorgverleners, medewerkers op naam, medewerkers niet op naam en voor servers verstrekt voor gebruik in de Zorgsector. In de certificaten van zowel UZI-passen als –server- certificaten wordt de identiteit van de houder, het UZI-nummer en aanverwante informatie opgenomen. Momenteel is generatie (G3) van UZI-register CA's in gebruik. De G3 generatie CA's zijn ondertekend met SHA-256 en sluit aan bij G3 van PKIoverheid.

De gangbare *technische vorm* van certificaten zijn X.509 versie 3 certificaten. X.509v3 is een specificatie van de ITU, zie [X509]. In een X.509 certificaat zijn technische en organisatorische kenmerken opgenomen en wordt er een koppeling gelegd tussen de identiteit van een actor en een publieke sleutel.

[2] Niet te verwarren met de G2 van het UZI-register, deze sluit aan bij de ***eerste*** generatie van PKIoverheid.

Verder is het mogelijk om naast standaard attributen extra kenmerken in het certificaat op te nemen, die door de uitgever als X.509v3 extensies in het certificaat worden gezet.

Attribuut	**Betekenis**
Serial number	Een serie nummer van het certificaat. Het betreft een serie nummer van de uitgevende CA. Dit nummer moet uniek zijn per *issuer*, maar hoeft niet oplopend te zijn.
Signature Algorithm	Dit geeft aan met welk cryptografisch algoritme dit certificaat door de uitgever is ondertekend.
Issuer	Dit geeft de DN (*Distinguished Name)* van de uitgever van het certificaat aan. Dit geeft ook gelijk aan wie dit certificaat heeft ondertekend. Deze komt overeen met het subject in het CA certificaat van de uitgever.
Validity	Met twee datums wordt het interval aangegeven, waarbinnen het certificaat geldig is. De tijd is normaal opgegeven in UTC, tenzij een expliciete tijdzone is opgenomen.
Subject	De DN van de houder aan wie het certificaat behoort. Dit geeft de identiteit aan, die aan de publieke sleutel is gekoppeld.
Subject Public Key Info	Dit bevat de benodigde informatie over de publieke sleutel. Het beschrijft het algoritme voor de sleutel (bv. RSA) en de specifieke numerieke delen van die sleutel.
Signature	Dit bevat de ondertekening van het certificaat. Deze wordt gedaan door de uitgever, en kan worden gecontroleerd met de publieke sleutel in het certificaat van de uitgever.

Tabel Mitz.STK.t3110 – X.509v3 attributen

Zoals eerder vermeld, wordt in de certificaten van zowel UZI-passen als –server- certificaten de identiteit van de houder opgenomen. Dit gebeurt in de `subjectAltName` extensie als een `othername`, met een vaste indeling.

De waarde hierin is een ASN.1 IAString met een OID (Object Identifier) van `2.5.5.5`. De string heeft de volgende vaste notatie:

`<OID CA>-<versie-nr>-<UZI-nr>-<pastype>-<Abonnee-nr>-<rol>-<AGB-code>`

Veld	Betekenis
OID CA	Dit is de OID van de CA en wijzigt niet per generatie en is generiek per type pas.
versie-nr	1 voor alle UZI-register certificaten.
UZI-nr	Het UZI-nummer.
Pastype	Een codering met 1 karakter. Voor het type certificaat, of pastype, bestaan een aantal standaard waarden: Z (Zorgverlener), N (Medewerker op naam), M (Medewerker niet op naam) en S (Servercertificaat).
Abonnee-nr	Voor UZI passen is dit het URA (UZI-Register Abonnee) nummer.
Rol	Aanduiding van een medisch beroep. Wanneer er geen rol is (servers, medewerkers) wordt op het certificaat rolcode 00.000 gebruikt.
AGB-code	De AGB-code van de abonnee (zorgaanbieder) van deze server. Deze wordt binnen Mitz niet gebruikt.

Tabel Mitz.STK.t3120 – X.509v3 extensie subjectAltName.otherName

Het certificaat dat gebruikt wordt voor het ondertekenen van het security token moet een PKIo-servercertificaat zijn. Het certificaat waarmee het security token ondertekend wordt betreft de publieke sleutel. De bijbehorende privé sleutel wordt gebruikt om de handtekening te genereren.

Het servercertificaat dat gebruikt wordt voor het ondertekenen van het token moet beschikken over de juiste sleuteltypes. Hiervoor wordt de X.509 extensie keyUsage gebruikt.

Naam	Sleuteltype (KeyUsage)	Hexadecimaal gebruik
authenticiteitcertificaat	digitalSignature	0x80

Tabel Mitz.STK.t3130 – X.509v3 extensie keyUsage

Elk security token dient getekend te worden met het juiste certificaat. Dit wordt verder in de verschillende beveiligingsimplementatiehandleidingen beschreven.

De ontvanger van het security token dient te controleren of het certificaat met de juiste `keyUsage` gebruikt is.

De ontvanger dient te controleren of het certificaat door de juiste CA is uitgegeven en geldig is ondertekend. De keten dient gecontroleerd te worden tot een vertrouwd certificaat (bijvoorbeeld een root certificaat van het UZI-register waarvan de keten eerder vertrouwd is).

De ontvanger dient te controleren of het certificaat niet verlopen of ingetrokken is, inclusief de certificaten in de keten tot een vertrouwd certificaat.

Om de handtekening te verifiëren, moet de ontvanger over de bijbehorende publieke sleutel beschikken. De ondertekenaar kan deze op verschillende manieren aan de ontvanger ter beschikking stellen:

1. door het certificaat met de publieke sleutel mee te zenden als binaire security token (BinarySecurityToken) of als sleutelinformatie (keyInfo).
2. door een verwijzing naar het certificaat mee te zenden; de ontvanger moet deze dan met bijvoorbeeld het LDAP protocol ophalen. Of, alvorens aan te sluiten, moeten zender en ontvanger de certificaten uitgewisseld hebben.

De opties worden in de volgende paragrafen in detail verder uitgewerkt. De eerste optie maakt de ontvanger flexibel. Als namelijk de afzender met een nieuw certificaat komt, hoeft de ontvanger niet zijn administratie voor zijn certificaten (store) bij te werken. De tweede optie heeft als voordeel dat de grootte van de berichten kleiner is.

Afhankelijk van de toepassing van het security token, wordt voor een optie gekozen.

In het testtraject worden testpassen gebruikt. Het gebruik hiervan wordt verder niet uitgewerkt in deze handleiding. De opbouw en werking is identiek, al staat de hiërarchie los van PKIoverheid.

4.4.1 CERTIFICAAT MEEZENDEN ALS BINARYSECURITYTOKEN

```
<wss:Security ...>
  <wss:BinarySecurityToken
    wsu:Id="signing-cert"
    ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-
1.0#X509v3"
    EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-
security-1.0#Base64Binary">MIIFd...PZaIvdgXOQ==
  </wss:BinarySecurityToken>
  <Signature ...>
    <SignedInfo ...>...</SignedInfo>
    <SignatureValue>...</SignatureValue>
    <KeyInfo>
      <wss:SecurityTokenReference>
        <wss:Reference URI="#signing-cert"
ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-
1.0#X509v3" />
      </wss:SecurityTokenReference>
    </KeyInfo>
  </Signature>
</wss:Security>
```

Een bespreking per punt:

```
<wss:Security ...>
```

Bovenstaand is de WSS Security header.

```
  <wss:BinarySecurityToken
    wsu:Id="signing-cert"
    ValueType=http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-
profile-1.0#X509v3
    EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-
message-security-1.0#Base64Binary">MIIFd...PZaIvdgXOQ==
  </wss:BinarySecurityToken>
```

Bij het opnemen van het certificaat wordt een BinarySecurityToken element opgenomen met het hele certificaat in Base 64 codering. De wsu:Id moet matchen met de referentie later, maar de invulling is verder vrij.

Wel moet deze waarde uniek zijn binnen het gehele document, aangezien het een ID is. De waarden voor ValueType en EncodingType zijn vast en verplicht.

```
  <Signature ...>
    <SignedInfo ...>...</SignedInfo>
    <SignatureValue>...</SignatureValue>
```

De Signature zoals beschreven.

```
    <KeyInfo>
      <wss:SecurityTokenReference>
```

De KeyInfo, met een verwijzing naar het BinarySecurityToken met het certificaat.

```
        <wss:Reference URI="#signing-cert"
ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-
profile-1.0#X509v3" />
```

De verwijzing in de URI moet kloppen met de wsu:Id in het BinarySecurityToken element. ValueType is vast en verplicht.

```
      </wss:SecurityTokenReference>
    </KeyInfo>
  </Signature>
</wss:Security>
```

Afsluiting van de elementen.

4.4.2 CERTIFICAAT MEEZENDEN ALS KEYINFO

```
<wss:Security ...>
  ...
  <Signature ...>
    <SignedInfo ...>...</SignedInfo>
    <SignatureValue>...</SignatureValue>
    <KeyInfo>
     <X509Data>
          <!-- Het certificaat -->
<X509Certificate>MIIFd...PZaIvdgXOQ==</X509Certificate>
         </X509Data>
    </KeyInfo>
  </Signature>
</wss:Security>
```

Een bespreking per punt:

```
<wss:Security ...>
```

Bovenstaand is de WSS Security header.

```
   <Signature ...>
    <SignedInfo>...</SignedInfo>
    <SignatureValue>...</SignatureValue>
```

De Signature zoals beschreven. De Signature en het daarbij gebruikte certificaat met de publieke sleutel kan als KeyInfo in het security token geïntegreerd zijn.

```
<KeyInfo>
```

```
  <X509Data>
   <!-- Het certificaat -->
   <X509Certificate>MIIFd...PZaIvdgXOQ==</X509Certificate>
  </X509Data>
</KeyInfo>
```

De KeyInfo, met het hele X509 certificaat in Base 64 codering.

```
  </Signature>
</wss:Security>
```

Afsluiting van de elementen.

4.4.3 CERTIFICAATVERWIJZINGEN

Bij het opnemen van een verwijzing naar het certificaat, wordt de volgende constructie opgenomen.

```
<KeyInfo>
  <wss:SecurityTokenReference>
    <X509Data xmlns="http://www.w3.org/2000/09/xmldsig#">
      <X509IssuerSerial>
        <X509IssuerName>CN=De Auteur CA,O=VZVZ,C=NL</X509IssuerName>
        <X509SerialNumber>359724...41160195</X509SerialNumber>
      </X509IssuerSerial>
    </X509Data>
  </wss:SecurityTokenReference>
</KeyInfo>
```

Een bespreking per punt:

```
<KeyInfo>
  <wss:SecurityTokenReference>
```

Het begin van de certificaatverwijzing. <KeyInfo> valt in de eerder gedeclareerde namespace van XML Signatures. <SecurityTokenReference> valt onder de WSS namespace, zie paragraaf **Fout! Verwijzingsbron niet gevonden.**.

```
<X509Data xmlns="http://www.w3.org/2000/09/xmldsig#">
```

De certificaatverwijzing zelf zit weer in de namespace van XML Signatures.

```
<X509IssuerSerial>
```

Begin van de certificaatgegevens.

```
<X509IssuerName>CN=De Auteur CA,O=VZVZ,C=NL</X509IssuerName>
```

De DN (Distinguished Name) van de CA (Certificate Authority). Dit geeft ook gelijk aan wie dit certificaat heeft ondertekend. Deze komt overeen met het subject in het CA certificaat van de uitgever. De attributen die in een DN van een security token zitten zijn CN (CommonName), O (OrganizationName) en C (CountryName), in die volgorde.

De attributen worden van de waarden gescheiden door een = teken, van elkaar gescheiden met komma's en opgenomen zonder spaties, met uitzondering van de spaties in de waarden zelf, zoals beschreven in [LDAP]. De attribuutwaarden van een DN worden per CA verder in de verschillende beveiligingshandleidingen uitgewerkt.

```
<X509SerialNumber>359724...41160195</X509SerialNumber>
```

Het decimale serialNumber van het certificaat, conform [X509CRL]. Het betreft een serie nummer van de uitgevende CA. Dit nummer is uniek per *issuer*, maar hoeft niet oplopend te zijn.

```
     </X509IssuerSerial>
   </X509Data>
  </wss:SecurityTokenReference>
</KeyInfo>
```

Afsluiting van de elementen.

5 OPBOUW VAN DE SECURITY HEADER

WS-Security is een OASIS standaard en biedt een normatief raamwerk voor de beveiliging van het berichtenverkeer aan. Voor het toepassen van WS-Security, moet er een SOAP Header in de SOAP Envelope worden opgenomen. Deze SOAP Header omvat de WS-Security informatie en de XML Signature.

```
<soap:Envelope ...>
  <soap:Header ...>
    <wss:Security ...>
      <!-- Security header-->
      ...
    </wss:Security>
  </soap:Header >
  </soap:Body >
    <!--HL7v3 (payload)-->
    ...
  </soap:Body >
</soap:Envelope >
```

Voor het toepassen van WS-Security in FHIR berichten wordt het security token in de HTTP-header geplaatst.

6 AFHANDELING VAN HET SECURITY TOKEN

6.1 VERIFICATIE EN CONTROLE

In de verschillende beveiligingsimplementatie handleidingen in Mitz kunnen aanvullende voorwaarden worden gesteld in het kader van verificatie en controle.

De Wbsn-z regelt dat voor berichten die betrekking hebben op een persoon waarvan het burgerservicenummer (BSN) bekend is, het BSN als gegeven in het security token op te nemen.

Bij ontvangst van het security token dient de ontvanger de volgende controles uit te voeren:

- voldoet de XML Signature aan de eisen die er in [XMLSIG] aan gesteld worden;
- voldoet het security token aan de eisen die er in dit document aan gesteld worden;
- valideert de signature over het security token conform de eisen die er in dit document aan gesteld worden;
- is het gebruikte certificaat correct getekend, evenals de certificaten tot een vertrouwd certificaat;
- is het gebruikte certificaat geldig, niet verlopen en niet ingetrokken;
- is er een match tussen het bericht en het security token conform de eisen die de (zorg)toepassing daaraan stelt.

Indien één van de controles niet bevestigd wordt, wordt een SOAP fout gegenereerd, zie paragraaf 6.2 Foutafhandeling. Indien een SOAP fout wordt gegenereerd *mag* het bericht door de ontvangende partij verder verwerkt worden wanneer de betreffende toepassing dat vereist.

6.2 FOUTAFHANDELING

De volgende SOAP fouten zijn van toepassing bij security tokens. Op de eerste SOAP foutmelding na, zijn ze onderdeel van WS-Security.

Omschrijving (faultstring)	**Faultcode**
Er is een (WS-Security) header aanwezig die niet bekend is, maar met mustUnderstand = "1". In dit geval moet deze Fault gegenereerd worden.	soap:MustUnderstand
An unsupported token was provided	wss:UnsupportedSecurityToken
An unsupported signature or encryption algorithm was used	wss:UnsupportedAlgorithm
An error was discovered processing the <wss:Security> header	wss:InvalidSecurity
An invalid security token was provided	wss:InvalidSecurityToken
The security token could not be authenticated or authorized	wss:FailedAuthentication
The signature or decryption was invalid	wss:FailedCheck
Referenced security token could not be retrieved	wss:SecurityTokenUnavailable
The message has expired	wss:MessageExpired

TABEL MITZ.STK.T3150

De volgende foutmeldingen worden daar aan toegevoegd.

Omschrijving (faultstring)	Faultcode
Authenticatietoken en bericht stemmen niet overeen	ao:AuthTokenMessageMismatch
Authenticatietoken is niet valide of compleet	ao:AuthTokenInvalid
Authenticatietoken buiten geldigheidsduur ontvangen	ao:ExpirationTimeError
Nonce is reeds gebruikt	ao:NonceRejected

TABEL MITZ.STK.T3160

De volgende statuscodes (URI referenties) worden door SAML onderkend en geretourneerd [SAML Core].

Een statuscode is een Uniforme Resource Name (URN) en is als volgt opgebouwd:

"urn:"<Namespace Identifier>":"<Name Specific String>

Voorbeeld:

"urn:oasis:names:tc:SAML:2.0:status:Success"

In de <statuscode> tabel Mitz.STK.t3170 wordt alleen de omschrijving en de specifieke string (<Name Specific String>) getoond. <StatusDetail> kan daarnaast gebruikt worden om een specifieke leesbare melding te retourneren (zie [SAML Core]).

Omschrijving	Name Specific String
The request could not be performed due to an error on the part of the requester.	Requester
The request could not be performed due to an error on the part of the SAML responder or SAML	Responder
The SAML responder could not process the request because the version of the request message was incorrect.	VersionMismatch
The responding provider was unable to successfully authenticate the principal.	AuthnFailed
The specified authentication context requirements cannot be met by the responder.	NoAuthnContext
The SAML responder or SAML authority is able to process the request but has chosen not to respond.	RequestDenied
The SAML responder or SAML authority does not support the request.	RequestUnsupported
The SAML responder cannot process any requests with the protocol version specified in the request.	RequestVersionDeprecated
The SAML responder cannot process the request because the protocol version specified in the request message is a major	RequestVersionTooHigh

Omschrijving	Name Specific String
upgrade from the highest protocol version supported by the responder.	
The SAML responder cannot process the request because the protocol version specified in the request message is too low.	RequestVersionTooLow
The resource value provided in the request message is invalid or unrecognized.	ResourceNotRecognized
The response message would contain more elements than the SAML responder is able to return.	TooManyResponses
The responding provider does not recognize the principal specified or implied by the request.	UnknownPrincipal
The SAML responder cannot properly fulfill the request using the protocol binding specified in the request.	UnsupportedBinding

TABEL MITZ.STK.T3170

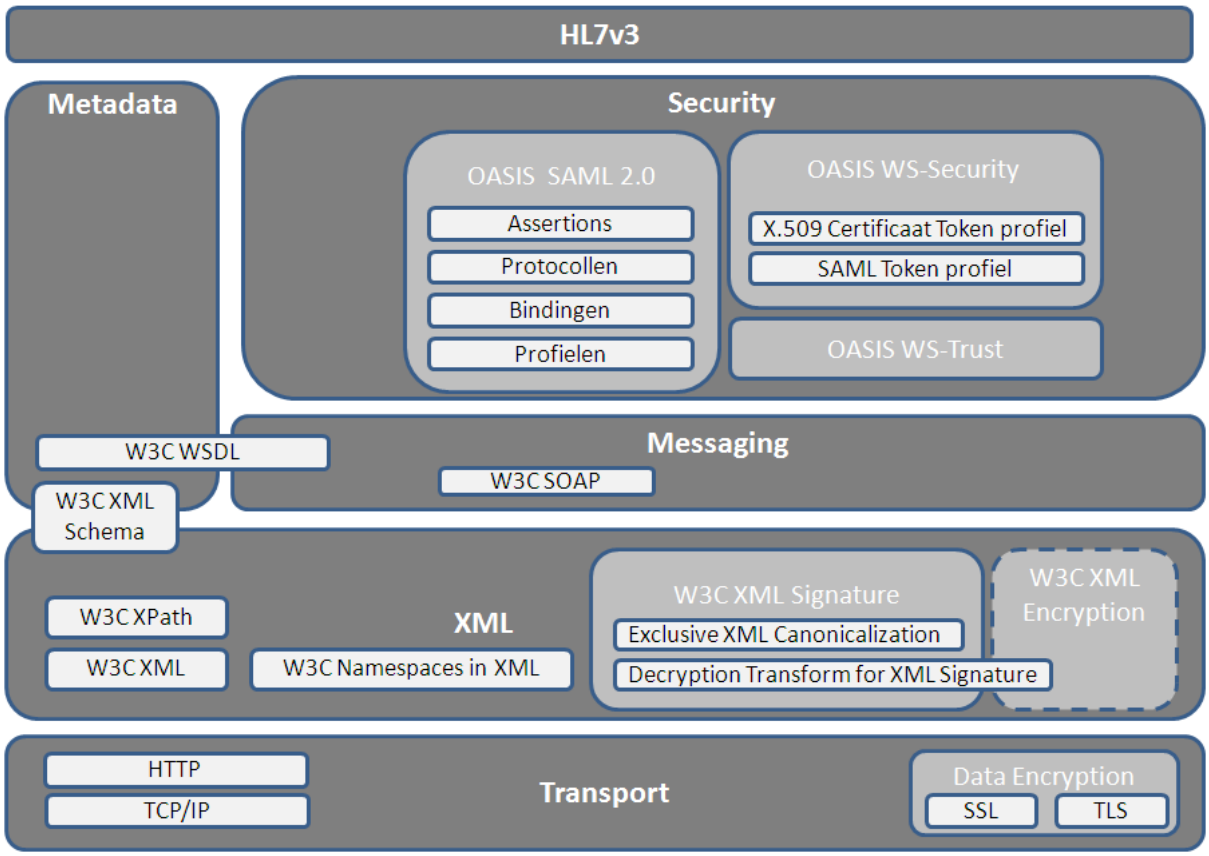
BIJLAGE A REFERENTIES

Referentie	Document	Versie
[BSP]	Basic Security Profile Version 1.0, Final. www.ws-i.org	1.0 30-mrt-2007
[EXCC14N]	Exclusive XML Canonicalization, Version 1.0, W3C Recommendation, 18 July 2002, http://www.w3.org/TR/xml-exc-c14n/	1.0 18-juli-2002
[HTTP]	RFC 2616 Hypertext Transfer Protocol -- HTTP/1.1 http://www.ietf.org	1.1
[LDAP]	Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names, IETF, Juni 2006, http://www.ietf.org/rfc/rfc4514.txt	juni-2008
[MIME]	Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies, www.ietf.org/rfc/rfc2045.txt	
[Namespaces]	Namespaces in XML 1.0 (Second Edition) http://www.w3.org/TR/xml-names/	1.0
[SAML Binding]	Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, March 2005 http://docs.oasis-open.org/security/saml/v2.0/saml-bindings-2.0-os.pdf	2.0 mrt-2005
[SAML Core]	Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, March 2005 http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf	2.0 mrt-2005
[SAML Profiles]	Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, March 2005 http://docs.oasis-open.org/security/saml/v2.0/saml-profiles-2.0-os.pdf	2.0 mrt-2005
[SAML Security]	Security and Privacy Considerations for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS Standard, March 2005 http://docs.oasis-open.org/security/saml/v2.0/saml-sec-consider-2.0-os.pdf	2.0 mrt-2005
[SOAP]	Simple Object Access Protocol (SOAP) 1.1 http://www.w3.org/TR/SOAP	1.1
[UZI CA Model]	CA Model, Pasmodel, Certificaat- en CRL-profielen Zorg CSP. http://www.uziregister.nl/ https://www.uziregister.nl/overhetuziregister/hierarchie-productieomgeving	7.0 16 jan 2014
[WSDL]	Web Services Description Language (WSDL) 1.1, W3C Note, 15 March 2001, http://www.w3.org/TR/wsdl	15-mrt-2001
[WSS]	Web Services Security: SOAP Message Security 1.0, OASIS Standard Specification, March 2004	mrt-2004

Referentie	Document	Versie
	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf	
[WS SAML Token Profile]	Web Service Security: SAML Token Profile 1.1 http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SAMLTokenProfile.pdf	1-feb-2006
[WS Soap Message Security]	WSS: SOAP Message Security 1.1 http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SOAPMessageSecurity.pdf	1-feb-2006
[WS Trust]	WS-Trust 1.4 http://docs.oasis-open.org/ws-sx/ws-trust/v1.4/os/ws-trust-1.4-spec-os.pdf	2-feb-2009
[WSX509]	Web Services Security X.509 Certificate Token Profile 1.0, OASIS Standard Specification, March 2004 http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0.pdf	mrt-2004
[X509]	Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks, Recommendation X.509, 08-2005 http://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-X.509-200508-I!!PDF-E&type=items	aug-2005
[X509CRL]	Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile http://www.ietf.org/rfc/rfc3280.txt	apr-2002
[XML]	Extensible Markup Language (XML) 1.0, W3C Recommendation, Fourth Edition, 16 August 2007 http://www.w3.org/TR/xml	16-aug-2007
[XMLC14N]	W3C Recommendation, "Canonical XML Version 1.0," 15 March 2001 http://www.w3.org/TR/xml-c14n	15-mrt-2001
[XMLSECURI]	Additional XML Security Uniform Resource Identifiers (URIs) http://www.ietf.org/rfc/rfc4051.txt	
[XMLSIG]	XML-Signature Syntax and Processing (Seconde Edition), W3C Recommendation, 12 February 2002 http://www.w3.org/TR/xmldsig-core/	10-juni-2008
[XPATH]	W3C Recommendation, "XML Path Language", 16 November 1999 http://www.w3.org/TR/xpath/	16-nov-1999

BIJLAGE B TECHNOLOGIEËN EN STANDAARDEN

Deze bijlage geeft een overzicht van de gebruikte technologieën en standaarden en beschrijft die profielen die nodig zijn om berichten met security tokens uit te rusten, voor de uitwisseling van beveiligingsgerelateerde informatie tussen vertrouwde partijen en die voldoen aan de eisen van authenticiteit.



Figuur 2: Technologieën en standaarden

6.1 XML SIGNATURE

XML Signature [XMLSIG] beschrijft de mogelijkheid om XML documenten of delen hiervan te ondertekenen. Een deel van de XML wordt gesigneerd (als tekst) en de signature wordt als XML bestand (Signature element) toegevoegd aan het XML document.

6.2 XML CANONICALIZATION

XML Canonicalization [XMLC14N] (kortweg C14N genaamd) is een methode om ervoor te zorgen dat XML altijd in dezelfde syntactische vorm aanwezig is. Met XML is het mogelijk dat software "onderweg" de vorm aanpast. Zo is in XML bijvoorbeeld per definitie `<tag/>` equivalent aan `<tag></tag>`. Voor een signature is dit echter niet hetzelfde. Met C14N wordt een XML uitdrukking in een canonieke vorm gebracht. Daarna kan dan op eenduidige wijze de signature gegenereerd worden. De ontvanger kan dan ook vóór het verifiëren van de ondertekening de canonieke vorm genereren, zodat het juiste – canonieke – document wordt vergeleken met de signature.

Er zijn twee vormen van canoniek maken: Inclusive en Exclusive [EXCC14N]. De tweede vorm is de meest gebruikte voor Web Services. Het verschil tussen Inclusive en Exclusive Canonicalization zit in de wijze waarop namespace declaraties worden meegenomen.

Bij Inclusive Canonicalization worden namespace declaraties uit een inkapselend document gewoon meegenomen: dat betekent dat de signature verandert wanneer een document ingekapseld wordt. Dat maakt Inclusive Canonicalization ongeschikt voor ingekapselde documenten, zoals een HL7v3 bericht in een SOAP Envelope. Daarmee is Exclusive Canonicalization de gebruikte optie binnen Mitz.

Voorbeeld van een bericht vóór het canoniek maken:

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <MFMT_IN002101 xmlns="urn:hl7-org:v3"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="urn:hl7-org:v3 ../schemas/MFMT_IN002101.xsd">
      <id extension="34234" root="2.16.528.1.1007.3.2.700222.1"/>
      <creationTime value="20040417161000"/>
      <versionCode code="NictizEd2005-Okt"/>
      <interactionId extension="MFMT_IN002101" root="2.16.840.1.113883.1.6"/>
      <profileId root="2.16.840.1.113883.2.4.3.11.1" extension="810"/>
      <processingCode code="P"/>
      <processingModeCode code="T"/>
      <acceptAckCode code="AL"/>
```

Hetzelfde bericht canoniek, exclusive zonder commentaar:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <MFMT_IN002101 xmlns="urn:hl7-org:v3" xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
xsi:schemaLocation="urn:hl7-org:v3 ../schemas/MFMT_IN002101.xsd">
      <id extension="34234" root="2.16.528.1.1007.3.2.700222.1"></id>
      <creationTime value="20040417161000"></creationTime>
      <versionCode code="NictizEd2005-Okt"></versionCode>
      <interactionId extension="MFMT_IN002101" root="2.16.840.1.113883.1.6"></interactionId>
      <profileId extension="810" root="2.16.840.1.113883.2.4.3.11.1"></profileId>
      <processingCode code="P"></processingCode>
      <processingModeCode code="T"></processingModeCode>
      <acceptAckCode code="AL"></acceptAckCode>
```

Te zien valt dat:

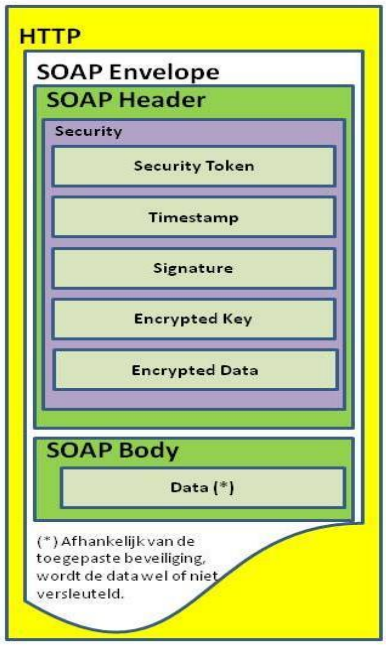
- de XML declaratie is verwijderd;
- attributen per element alfabetisch zijn gesorteerd;
- namespace declaraties voor een element in lexicografische volgorde eerder komen dan andere attributen van dat element;
- lege elementen een eind-tag krijgen.

Whitespace – spaties, tabs, regeleindes – zijn significant in XML. XML is immers begonnen als opmaaktaal voor documenten, en in een document is whitespace relevant. Canoniek maken laat de whitespace dus ongemoeid, behalve newlines die bij canoniek als een linefeed worden opgenomen.

6.3 WS-SECURITY

Web Services Security (WS-Security) is een OASIS (Organization for the Advancement of Structured Information Standards) standaard die de integriteit, authenticiteit en vertrouwelijkheid van gegevens op *bericht niveau* garandeert (en koppelt daardoor de beveiliging los van het gebruikte communicatie kanaal).

WS-Security biedt tevens een mechanisme voor het gebruik van WS-Security tokens om deze aan berichten te koppelen. In WS-Security is een WS-Security token een XML structuur met beveiligingsgegevens. Omdat WS-Security geen specifieke type WS-Security tokens vereist, gebruiken we in de rest van dit document de term "security token" voor "WS-Security token".



FIGUUR 3: SOAP BERICHT MET WS-SECURITY

WS-Security voegt zijn gegevens toe aan de SOAP header door een `<Security>` element te definiëren waarbinnen een aantal beveiligingsaspecten worden geregeld, zoals:

- Vertrouwelijkheid. Berichten kunnen (deels) versleuteld worden. Hiervoor wordt de XML Encryption (W3C) specificatie geïmplementeerd.
- Integriteit. Berichten kunnen digitaal ondertekend worden zodat zowel de zendende als de ontvangende partij ervan zijn verzekerd dat de inhoud niet door derden is gewijzigd. Hiervoor wordt de XML Signature specificatie (W3C) geïmplementeerd.
- Geldigheidsduur. Berichten kunnen voor een bepaalde duur geldig en bruikbaar zijn.
- Authenticatie en autorisatie. In de SOAP header(s) kunnen verschillende type WS-Security tokens gebruikt worden, zoals:
    - gebruikersnaam en wachtwoord van een gebruiker in tekstformaat;
    - X.509 certificaat voor de authenticatie van gebruikers;
    - SAML Assertion (bewering) over entiteiten.

WS-Security kent diverse profielen die het gebruik van de verschillende security tokens (WS-Security tokens) beschrijft en hoe deze in de SOAP header geplaatst moeten worden.

In de volgende paragrafen wordt WS-Trust als uitbreiding van WS-Security besproken en de WS-Security profielen X.509 Certificaat Token en SAML Token.

Er zijn twee versies van WS-Security, 1.0 [WSS] en 1.1 [WS Soap Message Security], verschenen in 2004 respectievelijk 2006. De 1.1-specificatie benadrukt verbeteringen aan "security token" ondersteuning, berichtbijlagen en het beheer van rechten.

6.3.1 WS-TRUST

WS-Trust [WS Trust] is eveneens een OASIS standaard en definieert uitbreidingen op WS-Security voor het verkrijgen en valideren van security tokens en voor het vormgeven van trust relaties. De reden voor het bestaan van WS-Trust is dat WS-Security alleen niet goed schaalt in een complexe omgeving, zoals Mitz.

Indien er meerdere applicaties bestaan, verdeeld over verschillende security domeinen is het moeilijk om security tokens op de juiste manier te verkrijgen en te valideren. WS-Trust voorziet in een ontkoppelpunt voor het valideren van security tokens en het uitgeven of transformeren van deze security tokens in een bepaalde vorm (bijvoorbeeld een SAML token uit het ene domein transformeren in een SAML token uit een ander domein). Dit ontkoppelpunt heet "Security Token Service" (STS).

Binnen Mitz wordt gebruik gemaakt van WS-Trust 1.4.

6.3.2 X.509 CERTIFICATE TOKEN PROFILE

Dit profiel [WSX509] beschrijft het gebruik van X.509 certificaat als "security token" in WS-Security. Dit profiel definieert de opties voor het invoegen van een X.509 certificaat en gerelateerde gegevens. Hiervan zijn voor ons relevant:

- X.509 certificaat in binaire vorm (geserialiseerd na Base64 encoding)
- Verwijzing naar X.509 certificaat aan de hand van de naam van de uitgevende CA en het serienummer van het certificaat binnen die CA (X509IssuerSerial).

Binnen Mitz wordt gebruik gemaakt van WS-Security X.509 profiel 1.0.

6.3.3 SAML TOKEN PROFILE

Dit profiel [WS SAML Token Profile] beschrijft het invoegen van SAML assertions als "security token" in WS-Security voor SAML v2.0. Hiervan is van belang hoe de SAML assertion in de header geplaatst moet worden:

- Attaching Security Tokens met SAML v2.0

6.4 SAML

Security Assertion Markup Language (SAML) [SAML Core] is een op XML gebaseerde OASIS standaard voor de uitwisseling van beveiligingsgerelateerde informatie.

De uitwisseling van de beveiligingsgerelateerde informatie gebeurt op basis van assertions. Een assertion is een uitspraak of bewering van een vertrouwde autoriteit (partij die verantwoordelijk is voor de waarmerking van die uitspraak) over een gebruiker en wordt door de Service Provider (partij die een applicatie of bron beschikbaar stelt voor de gebruiker) gebruikt om te bepalen of de gebruiker toegang mag krijgen tot een opgevraagde dienst of informatie. Uitspraken die een assertion kunnen bevatten zijn op te delen in een drietal categorieën:

- Authenticatie uitspraken: Deze uitspraken vertellen iets over het feit of de gebruiker zijn identiteit eerder succesvol kenbaar heeft gemaakt bij een Identity Provider. Voorbeelden van authenticatie assertions zijn: "Dit is arts X" en "Op datum D en tijdstip T heeft patiënt Y zich geauthenticeerd bij DigiD".
- Autorisatie uitspraken: Een dergelijke uitspraak vertelt tot welke delen van aangeboden diensten en/of informatie een gebruiker toegang heeft. In dit geval vertrouwt de dienstverlener volledig op de Identity Provider voor fijnmazige afscherming van de dienst, in plaats van zelf autorisatie bepalingen te maken.

Een voorbeeld van een autorisatie uitspraak: "Patiënt P heeft toegang tot dossier D". Voor patiëntauthenticatie wordt geen gebruik gemaakt van deze uitspraken.
- Attribuut uitspraken: Een attribuut uitspraak geeft aan wat de kenmerken van een gebruiker zijn, zoals die bekend zijn bij een Identity Provider. Een voorbeeld van een attribuut uitspraak: "Patiënt P heeft burgerServiceNummer 123456789".

Verder beschrijft de SAML standaard onder andere:
- Verschillende mechanismen (protocollen) voor het opvragen en het verkrijgen van assertions, authenticatie verzoek, artifacts, name identifiers en logout verzoek [SAML Core];
- Bindingen met transport protocollen zoals HTTP en SOAP [SAML Binding];
- Standaard manieren (profielen) voor gebruik van SAML [SAML Profiles]

SAML heeft de WS-Security standaard geadopteerd. Met behulp van het SAML Token profile is het mogelijk om één of meerdere assertions als "security token" in SOAP berichten mee te sturen en digitaal te ondertekenen.

Binnen Mitz wordt gebruik gemaakt van HTTP over TLS 1.2 om de vertrouwelijkheid van de assertions te waarborgen [SAML Security].

In de volgende paragrafen worden SAML2.0 profielen [SAML Profiles] beschreven, met referenties aan de OASIS SAML 2.0 specificaties.

6.4.1 SSO (Single Sign-On) Profile

Het Single Sign-On profiel (SSO) [SAML Profiles] is een SAML profiel dat aangeeft hoe gebruik te maken van het SAML authenticatie vraag en antwoord protocol in combinatie met verschillende SAML-bindings, zoals SOAP en HTTP. De communicatie bij authenticatie gaat tussen een Identity Provider (partij die verantwoordelijk is voor de waarmerking van de gebruiker) en een Service Provider (partij die een applicatie of bron beschikbaar stelt voor de gebruiker).

Voor het SSO profiel zijn de volgende twee aspecten van belang:
1. Bepalen of de berichten door de Identity Provider of door de Service Provider worden geïnitieerd.
2. Bepalen welke binding er gebruikt wordt voor de berichtuitwisseling tussen Identity Provider en Service Provider.

Het eerste aspect heeft te maken met waar de gebruiker start met het SSO proces voor de berichtuitwisseling. SAML ondersteunt twee algemene berichtstromen ter ondersteuning van het SSO proces. Binnen Mitz initieert de Service Provider het SSO profiel.

Het tweede aspect heeft te maken met de verschillende SAML bindingen (HTTP, SOAP of Artefact), die gebruikt worden tussen het uitwisselen van informatie tussen de Identity Provider en de Service Provider. Binnen Mitz maken we gebruik van artifacts, die op SOAP/HTTP gebaseerd zijn.

Voor het SSO profiel zijn de volgende twee SAML berichten van belang; een verificatieverzoek bericht van de Service Provider naar een Identity Provider en het antwoord daarop dat van de Identity Provider naar de Service Provider wordt verzonden.

6.4.2 Artifact Resolution Profile

Het Artefact Resolution profiel [SAML Profiles] is een SAML profiel dat aangeeft hoe gebruik te maken van SAML artefacten in combinatie met SAML-bindings. De communicatie vindt plaats tussen de artefact opvrager (Service Provider) en de artefact resolution service (Identity Provider).

De artefact opvrager initieert het profiel door een vraag over een artefact naar de artefact resolution service te versturen. De artefact resolution service reageert hierop door (altijd) antwoord op de vraag te geven.

Een SAML artefact is een klein gestructureerd data object van een vast formaat dat verwijst naar een groter SAML protocol bericht [SAML Binding], bijvoorbeeld een SAML assertion. Een SAML artefact wordt in URL ingebed en overgebracht via een HTTP-bericht.