e-Invoice Document Signature Creation Guideline

REVISION CONTROL AND CHANGE HISTORY

Revision Number	Approval Date	Approved By
Revision 1.0	24th May 2024	Nazril Bin Mohd Ghani
Revision 1.1	5th July 2024	Nazril Bin Mohd Ghani
Revision 1.2	29th July 2024	Nazril Bin Mohd Ghani

Disclaimer

The information in this document is intended only for the person or entity to which it is addressed and may contain confidential and/or privileged material. This guideline is created based on e-Invoice Guideline made available by LHDNM at their portal https://sdk.myinvois.hasil.gov.my/signature-creation.

Any review, retransmission, dissemination, or other use of, or taking of any action in reliance upon this information by persons or entities other than the intended recipient is prohibited. If you received this document in error, please contact the sender and delete the material from all computers or medium.

Ownership

This is the intellectual property of Pos Digicert Sdn. Bhd. and all its components belong to Pos Digicert Sdn. Bhd, located in Star Central, Cyberjaya.

Sample Code

The sample code in this document is provided “AS IS” and any express or implied warranties, including the implied warranties of merchantability and fitness for a particular purpose are disclaimed. We voluntarily put forth our effort to enhance the industry by providing this sample code and Pos Digicert will not be held liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) sustained by the client or a third party, however caused and on any theory of liability, whether in contract, strict liability, or tort arising in any way out of the use of this sample code, even if advised of the possibility of such damage.

Feedback

To get further clarification on the API/web service or the usage of the API/web service, email can be sent to einvoice@posdigicert.com.my

Confidentiality Statement

Contents

1.0 High-Level Process

The overall steps to follow when preparing a single document for submission are:

- **Step 1:** Create document XML or JSON (no signature elements yet) for a single document
- **Step 2:** Apply transformations to the document
- **Step 3:** Canonicalize the document and generate the document hash (digest) (property reference **DocDigest**)
- **Step 4:** Sign the document digest (property reference **Sig**)
- **Step 5:** Generate the certificate hash (Digest) (property reference **CertDigest**)
- **Step 6:** Populate the signed properties section
- **Step 7:** Generate Signed Properties Hash (Digest) (property reference **PropsDigest**)
- **Step 8:** Populate the information in the document to create the signed document

1.1 Step 1: Generate XML/JSON Document in UBL 2.1

- Create document XML or JSON (no signature elements yet) for a single document.
- XML document must be in format UBL 2.1.
- Pos Digicert is not responsible to provide a guideline on UBL 2.1 since there are many resources on the Internet on how to perform this.

Reference:

XML	https://docs.oasis-open.org/ubl/UBL-2.1.html
JSON	https://docs.oasis-open.org/ubl/UBL-2.1-JSON/v2.0/cnd01/UBL-2.1-JSON-v2.0-cnd01.html

1.2 Step 2: Apply transformations to the document

- Make sure the source document is in UTF-8 format.
- Remove the XML version.
- Remove the not required elements if any exists (UBLExtension (XPath: [local-name()='Invoice']//[local-name()='UBLExtensions']), and Signature (XPath: [local-name()='Invoice']//[local-name()='Signature']))
- Remove XML element UBL Extension below from the XML document, if any:

```
<ext:UBLExtension>
<ext:ExtensionURI>urn:oasis:names:specification:ubl:dsig:enveloped:xades</ext:ExtensionURI>
<ext:ExtensionContent>
<!--  Please note that the signature values are sample values only  -->
<sig:UBLDocumentSignatures xmlns:sig="urn:oasis:names:specification:ubl:schema:xsd:CommonSignatureComponents-
2" xmlns:sac="urn:oasis:names:specification:ubl:schema:xsd:SignatureAggregateComponents-
2" xmlns:sbc="urn:oasis:names:specification:ubl:schema:xsd:SignatureBasicComponents-2">
<sac:SignatureInformation>
<cbc:ID>urn:oasis:names:specification:ubl:signature:1</cbc:ID>
<sbc:ReferencedSignatureID>urn:oasis:names:specification:ubl:signature:Invoice</sbc:ReferencedSignatureID>
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#" Id="signature">
<ds:SignedInfo>
<ds:CanonicalizationMethod Algorithm="http://www.w3.org/2006/12/xml-c14n11"/>
<ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
<ds:Reference Id="id-doc-signed-data" URI="">
<ds:Transforms>
<ds:Transform Algorithm="http://www.w3.org/TR/1999/REC-xpath-19991116">
<ds:XPath>not(//ancestor-or-self::ext:UBLExtensions)</ds:XPath>
</ds:Transform>
<ds:Transform Algorithm="http://www.w3.org/TR/1999/REC-xpath-19991116">
<ds:XPath>not(//ancestor-or-self::cac:Signature)</ds:XPath>
</ds:Transform>
<ds:Transform Algorithm="http://www.w3.org/2006/12/xml-c14n11"/>
</ds:Transforms>
<ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
<ds:DigestValue>RvCSpMYz8009KbJ3ku72oaCFWpzEfQNcpc+5bulh3Jk=</ds:DigestValue>
</ds:Reference>
<ds:Reference Type="http://www.w3.org/2000/09/xmldsig#SignatureProperties" URI="#id-xades-signed-props">
<ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
<ds:DigestValue>OGU1M2Q3NGFkOTdkYTRiNDVhOGZmYmU2ZjE0YzI3ZDhhNjlmM2EzZmQ4MTU5NTBhZjBjNDU2MW
ZlNjU3MWU0ZQ==</ds:DigestValue>
</ds:Reference>
</ds:SignedInfo>
<ds:SignatureValue>MEYCIQDYsDnviJYPgYjyCIYAyzETeYthIoJaQhChblP4eAAPPAIhAJl6zfHgiKmWTtsfUz8YBZ8QkQ9r
BL4Uy7mK0cxvWooH</ds:SignatureValue>
<ds:KeyInfo>
<ds:X509Data>
<ds:X509Certificate>MIIEQzCCAyugAwIBAgIhAOkUChItLeodmoK/A7B0XLcSUCvT4jgrSeYBOeZ1G8VPMA0GCSqGSIb3D
QEBBQUAMIG9MQswCQYDVQQGEwJLTDEhMB8GA1UECgwYQ29udG9zbyBNYWxheXNpYSBTZG4gQmhkMSEwHwY
DVQQLDBhDb250b3NvIE1hbGF5c2lhIFNkbiBCaGQxITAfBgNVBAMMGENvbnRvc28gTWFsYXlzaWEgU2RuIEJoZDEiMC
AGCSqGSIb3DQEJARYTbm9lbWFpbEBjb250b3NvLmNvbTEhMB8GA1UEAwwYQ29udG9zbyBNYWxheXNpYSBTZG4gQ
mhkMB4XDTI0MDQwMzA5NTM1MFoXDTI3MDQwNDA5NTM1MFowgZoxCzAJBgNVBAYTAktMMSEwHwYDVQQKDBhD
b250b3NvIE1hbGF5c2lhIFNkbiBCaGQxITAfBgNVBAsMGENvbnRvc28gTWFsYXlzaWEgU2RuIEJoZDEhMB8GA1UEAwwY
Q29udG9zbyBNYWxheXNpYSBTZG4gQmhkMSIwIAYJKoZIhvcNAQkBFhNub2VtYWlsQGNvbnRvc28uY29tMIIBIjANBgkqh
kiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA8yRxOigcbDzObxhNmVklzyJOItz4eSZHUv+JEy7nTosOg/wcFcDgrJDw6LZ/Mr
6aW98VJ930hlSw52fOPiMnXTyJLF6ZjISPsTjlrn9eKnbWt6DWqFIDWDIaXVhAcfFRkKftSFgSIEO9NIb+kmV6K/LotWEiLglz6
KZE3EopSF+pXa1LmwC8v0UhK8V8LcxqIe3dBq/jzyaWsLx5D4zdQqBSFEXrfp0A+N/93uAtQa35Fj3ypEpSzF/EQ6bDO/Gw
BKQm3lCny6AJB/l/kCbC/X+oMxkTOo9zW5hcdRiqmAa4iIrhORIOTlj5qfnngjuITnSMdK5kXSLJxC6SDlKtVwIDAQABo08wTT
AdBgNVHQ4EFgQU2pvYc/z5Prqp8Dt8PM1C/df1ysQwHwYDVR0jBBgwFoAU2pvYc/z5Prqp8Dt8PM1C/df1ysQwCwYDVR0
RBAQwAoIAMA0GCSqGSIb3DQEBBQUAA4IBAQBHzVHoQk+cAwunDlmrBjWYxfzmF3Adab81HKug+riDGiSG3bNntAwRV
kDC4onG680Ucsuhxeyj18gkAtR/5ZWu3RDZwcYoBMuQzUSS9U5bwg5VqCqxEfTQCSERjuCa8lt99EcgY06e8a8WEwcY19
LKVVwtrTJnlHvXhmcheumX3pfjPb5u0c0WKnbkj5mow75TuEmc0k1qow6Z6H5O6cPhX+eyNQSFZ3QnC0W2oIZTi96TVT4J
H8LOPurZ5AdG9maQNIypaZ0gYPtAJISP+nxPOHmloicecdLLaMG/cvDf2+/bJR2P98dTuTZqgKrvWHkiMOma62MVx5dbcm
bxgxU6</ds:X509Certificate>
</ds:X509Data>
</ds:KeyInfo>
<ds:Object>
<xades:QualifyingProperties xmlns:xades="http://uri.etsi.org/01903/v1.3.2#" Target="signature">
<xades:SignedProperties Id="id-xades-signed-props">
<xades:SignedSignatureProperties>
<xades:SigningTime>2024-04-01T00:41:21Z</xades:SigningTime>
<xades:SigningCertificate>
<xades:Cert>
<xades:CertDigest>
<ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
<ds:DigestValue>YTJkM2JhYTcwZTBhZTAxOGYwODMyNzY3NTdkZDM3YzhjY2IxOTIyZDZhM2RlZGJiMGY0NDUzZWJh
YWI4MDhmYg==</ds:DigestValue>
```

```
</xades:CertDigest>
<xades:IssuerSerial>
<ds:X509IssuerName>CN = Contoso Malaysia Sdn Bhd, E = noemail@contoso.com, OU = Contoso Malaysia Sdn Bhd, O = Contoso Malaysia Sdn Bhd, C = MY</ds:X509IssuerName>
<ds:X509SerialNumber>24753567</ds:X509SerialNumber>
</xades:IssuerSerial>
</xades:Cert>
</xades:SigningCertificate>
</xades:SignedSignatureProperties>
</xades:SignedProperties>
</xades:QualifyingProperties>
</ds:Object>
</ds:Signature>
</sac:SignatureInformation>
</sig:UBLDocumentSignatures>
</ext:ExtensionContent>
</ext:UBLExtension>
</ext:UBLExtensions>
```

1.3 Step 3: Canonicalize the document and generate the document hash (digest)

- Prepare document canonical version, to be used for signing:
  - XML format - apply xml-c14n11 canonicalization to the document
  - JSON format - minify the document by removing following elements from it:
    - Whitespaces
    - Line breaks
    - Comments

- Hash the canonicalized document invoice body using SHA-256.
  - Use a HEX-to-Base64 encoder to encode the hashed value and convert it to base 64.
  - The output will be set as the value of the property **DocDigest. Reference: https://sdk.myinvois.hasil.gov.my/signature/#document-signed-data**

(this space is intentionally left blank)

- Below is sample code using Java and Apache XML Security https://santuario.apache.org/download.html

```
/*************************************************************************
* 1. Canonicalize XML using c14n11
*************************************************************************/

String xml = "xml\\sample-xml-from-lhdn.xml";
String output = "xml\\sample-xml-from-lhdn-can.xml";
byte[] xmlBytes = Files.readAllBytes(Paths.get(xml));;

org.apache.xml.security.Init.init();
Canonicalizer canon =
            Canonicalizer.getInstance(Canonicalizer.ALGO_ID_C14N11_OMIT_COMMENTS
boolean secureValidation = true;

OutputStream os = new FileOutputStream(output);
canon.canonicalize(xmlBytes, os, secureValidation);
```

```
/*************************************************************************
* 2. Generate document hash
*************************************************************************/

String input = CalculateSHA256.readFile(output);

String hash = CalculateSHA256.calculateSHA256Base64(input.getBytes(StandardCharsets.UTF_8));
System.out.println("SHA-256 Hash in Base64: " + hash);


public static String calculateSHA256Base64(byte[] input) {
      try {
         // Create a MessageDigest instance for SHA-256
         MessageDigest digest = MessageDigest.getInstance("SHA-256");

         // Apply the hash function to the input string's bytes
         byte[] hashBytes = digest.digest(input);

         // Encode the byte array to a Base64 string
         String base64Hash = Base64.getEncoder().encodeToString(hashBytes);

         return base64Hash;

      } catch (NoSuchAlgorithmException e) {
         throw new RuntimeException(e);
      }
   }


            public static String readFile(String filePath) {
                        String content = null;
                        List<String> lines = null;
                        try {
                                    lines = Files.readAllLines(Paths.get(filePath));
                                    content = String.join("\n", lines);
                        } catch (IOException e) {
                                    e.printStackTrace();
                        }

                         return content;
            }
```

1.4 Step 4: Sign the document digest

- Sign the generated invoice hash with RSA-SHA256 using the signing certificate private key
- The output will be set as the value of the property Sig. Reference: https://sdk.myinvois.hasil.gov.my/signature/#signature-element
- Sample code to sign document hash using soft certificate refer to **POS DIGICERT SAMPLE CODE FOR SOFT CERT GUIDELINE rev 1.1**
- Sample code to sign document hash using roaming certificate refer to **POS DIGICERT ROAMING ORGANISATION CERT API GUIDELINE rev 1**

1.5 Step 5: Generate the certificate hash

- Hash the signing certificate using SHA-256
- Encode the certificate hash using Base64 encoding
- Sample code:

```
/**********************************************************************
                    * 3. Generate the certificate hash
                    **********************************************************************/
String certhash = CalculateSHA256.calculateSHA256Base64(cert.getEncoded());
System.out.println("SHA-256 cert Hash in Base64: " + certhash)
```

1.6 Step 6: Populate the signed properties section

Update the document, being signed, by filling-in the following properties as per the guidance below:

FIELD	VALUE	URL MAPPING PATH
DigestValue	Encoded certificate hashed property reference **CertDigest**	/Invoice /ext:UBLExtensions /ext:UBLExtension /ext:ExtensionContent /sig:UBLDocumentSignatures /sac:SignatureInformation /ds:Signature /ds:Object /xades:QualifyingProperties / xades:SignedProperties/xades:SignedSignature Properties / xades:SigningCertificate /xades:Cert /xades:CertDigest /ds:DigestValue

SigningTime	Sign timestamp as current datetime in UTC	/Invoice /ext:UBLExtensions /ext:UBLExtension /ext:ExtensionContent /sig:UBLDocumentSignatures /sac:SignatureInformation /ds:Signature /ds:Object /xades:QualifyingProperties /xades:SignedProperties /xades:SignedSignatureProperties /xades:SigningTime
X509IssuerName	Signing certificate issuer name	/Invoice /ext:UBLExtensions /ext:UBLExtension /ext:ExtensionContent /sig:UBLDocumentSignatures /sac:SignatureInformation /ds:Signature /ds:Object /xades:QualifyingProperties / xades:SignedProperties /xades:SignedSignatureProperties / xades:SigningCertificate /xades:Cert /xades:IssuerSerial /ds:X509IssuerName
X509SerialNumber	Signing certificate serial number	/Invoice /ext:UBLExtensions /ext:UBLExtension /ext:ExtensionContent /sig:UBLDocumentSignatures /sac:SignatureInformation /ds:Signature /ds:Object /xades:QualifyingProperties / xades:SignedProperties /xades:SignedSignatureProperties / xades:SigningCertificate /xades:Cert /xades:IssuerSerial /ds:X509SerialNumber

Now the signing properties section should be ready to be hashed. Sample code to get the values to be populated in the XML is available in the soft certificate and roaming certificate document guidelines.

- Sample code for soft certificate refer to **POS DIGICERT SAMPLE CODE FOR SOFT CERT GUIDELINE rev 1.1**
- Sample code for roaming certificate refer to **POS DIGICERT ROAMING ORGANISATION CERT API GUIDELINE rev 1**

1.7 Step 7: Generate Signed Properties Hash

- Get the properties tag only using the XPath (/Invoice/ext:UBLExtensions/ext:UBLExtension/ext:ExtensionContent/sig:UBLDocumentSignatures/sac:SignatureInformation/ds:Signature/ds:Object/xades:QualifyingProperties/xades:SignedProperties)
- Linearize the XML block (properties tag) and remove the spaces
- Hash the property tag using SHA-256.
- Encode the hashed property tag using HEX-to-Base64 Encoder
- The value generated would be the property named PropsDigest
- Sample code:

```
/**************************************************************************
                * 3. Generate the certificate hash
                **************************************************************************/
String certhash = CalculateSHA256.calculateSHA256Base64(cert.getEncoded());
System.out.println("SHA-256 cert Hash in Base64: " + certhash)
```

1.8 Step 8: Populate the information in the document to create the signed document

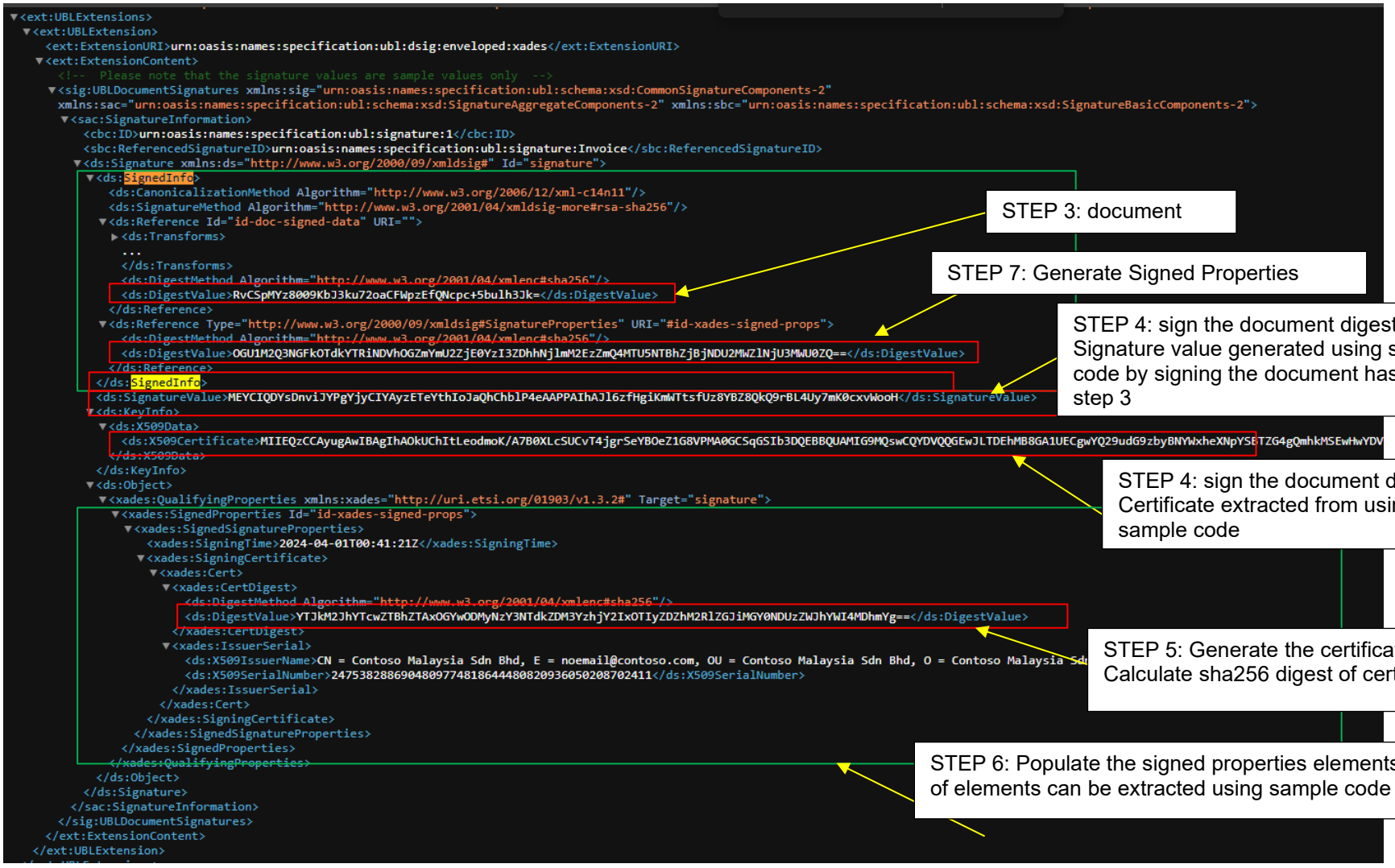
Populate the properties gathered into the document as per the below table to generate the signed document.

FIELD	VALUE	URL MAPPING PATH
SignatureValue	Digital Signature property ref **Sig**	/Invoice /ext:UBLExtensions /ext:UBLExtension /ext:ExtensionContent /sig:UBLDocumentSignatures /sac:SignatureInformation /ds:Signature /ds:SignatureValue
X509Certificate	Certificate	/Invoice /ext:UBLExtensions /ext:UBLExtension /ext:ExtensionContent /sig:UBLDocumentSignatures /sac:SignatureInformation /ds:Signature /ds:KeyInfo /ds:X509Data /ds:X509Certificate

DigestValue	Encoded signed Properties hash property reference **PropsDiges**	/Invoice /ext:UBLExtensions /ext:UBLExtension /ext:ExtensionContent /sig:UBLDocumentSignatures /sac:SignatureInformation /ds:Signature /ds:SignedInfo /ds:Reference[@URI='#id-xades-signed-props'] /ds:DigestValue
DigestValue	Encoded invoice hash property reference **DocDigest**	/Invoice /ext:UBLExtensions /ext:UBLExtension /ext:ExtensionContent /sig:UBLDocumentSignatures /sac:SignatureInformation /ds:Signature /ds:SignedInfo /ds:Reference[@Id='id-doc-signed-data'] /ds:DigestValue

The completed signed xml document as per sample at
https://sdk.myinvois.hasil.gov.my/files/one-doc-signed.xml

(this space is intentionally left blank)



2.0 List of Signature Error Code

CODE	DESCRIPTION
DS300	Failed to parse input document.
DS301	Signature not found
DS302	Signature count not equal to 1.
DS303	Signing time not found.
DS304	Signing time is not within validity period.
DS305	X509Certificate is not valid
DS306	SERIALNUMBER is empty in the certificate.
DS307	OI is empty in the certificate
DS308	Incompatible certificate for the Submission Channel. Organizational certificate cannot be used for Portal/Mobile submission.
DS309	Incompatible certificate for the Submission Channel. Personal certificate cannot be used for ERP submission.
DS310	Signer of invoice doesn't match the submitter of document. UserLoginId doesn't match with the SERIALNUMBER.
DS311	Signer of invoice doesn't match the submitter of document. TIN doesn't match with the OI.
DS312	Submitter registration/identity number doesn't match with the certificate SERIALNUMBER.
DS313	Certificate digest algorithm is doesn't match expected value of "http://www.w3.org/2001/04/xmlenc#sha256"
DS314	Document digest method doesn't match expected value of "http://www.w3.org/2001/04/xmlenc#sha256"
DS315	Signed properties digest method doesn't match expected value of "http://www.w3.org/2001/04/xmlenc#sha256"
DS316	Signature method doesn't match supported method of "http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"
DS317	Certificate digest is missing or empty.
DS318	Certificate digest value doesn't match digest calculated value from provided certificate.
DS319	Signed properties section digest is missing or empty.
DS320	Signed properties digest value doesn't match digest calculated value from provided signed properties section where ID is id-xades-signed-props.
DS321	Document digest is missing or empty
DS322	Document digest value doesn't match digest calculated value from existing document content. Document content might have changed.
DS323	Signature method used doesn't match the expected value of "urn:oasis:names:specification:ubl:dsig:enveloped:xades"

CODE	DESCRIPTION
DS324	Certificate X509IssuerName is missing or empty in the signed properties section.
DS325	Certificate X509SerialNumber is missing or empty in the signed properties section.
DS326	Certificate X509IssuerName doesn't match the X509IssuerName value provided in the signed properties section.
DS327	Cannot parse the certificate X509SerialNumber value from the signed properties section.
DS328	Certificate X509SerialNumber doesn't match the X509SerialNumber value provided in the signed properties section.
DS329	Certficate is not valid according to the chain of trust validation or has been issued by an untrusted certificate authority.
DS330	TIN provided cannot be found in Core system.
DS331	Failed to parse input document.
DS332	Document signature value is empty or doesn't exist.
DS333	Document signature value is not a valid signature of the document digest using the public key of the certificate provided.
DS334	Certificate cannot be used to validate the document signature value.
DS335	Failed to validate the document signature value.
DS336	Unable to parse Signing Time. Signing Time format is invalid.
DS337	Unable to parse certificate SERIALNUMBER. Certificate SERIALNUMBER is invalid.
DS338	Unable to parse certificate OI. Certificate OI is invalid.

(end of document)