Network Functions Virtualisation (NFV) Release 2; Protocols and Data Models; Network Service Descriptor File Structure Specification

Disclaimer

The present document has been produced and approved by the Network Functions Virtualisation (NFV) ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG. It does not necessarily represent the views of the entire ETSI membership.
Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for ETSI members and non-members, and can be found in ETSI SR 000 314: “Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards”, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

This Group Specification (GS) has been produced by ETSI Industry Specification Group (ISG) Network Functions Virtualisation (NFV).

Modal verbs terminology

In the present document "shall", "shall not", "should", "should not", "may", "need not", "will", "will not", "can" and "cannot" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"must" and "must not" are NOT allowed in ETSI deliverables except when used in direct citation.
1 Scope

The present document specifies the structure of the Network Service Descriptor (NSD) file archive and the naming conventions for the different files it contains, fulfilling the requirements specified in ETSI GS NFV-IFA 014 [1] for an NSD file structure.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference/.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.


NOTE: Available at http://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.2/csprd01/TOSCA-Simple-Profile-YAML-v1.2-csprd01.pdf.


NOTE: Available at https://www.iana.org/assignments/hash-function-text-names/hash-function-text-names.xhtml.


NOTE: Available at https://www.iana.org/assignments/media-types/media-types.txt.


[9] IETF RFC 3629: "UTF-8, a transformation format of ISO 10646".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.
The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] TOSCA-v1.0-os: "TOSCA Version 1.0".
[i.2] ETSI GS NFV 003: "Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV".
[i.3] ETSI GS NFV-SOL 001: "Network Functions Virtualisation (NFV) Release 2; Protocols and Data Models; NFV descriptors based on TOSCA specification".
[i.4] ETSI GS NFV-SOL 006: "Network Functions Virtualisation (NFV) Release 2; Protocols and Data Models; NFV descriptors based on YANG specification".

3 Definition of terms, symbols and abbreviations

3.1 Terms
For the purposes of the present document, the terms given in ETSI GS NFV 003 [i.2] apply.

3.2 Symbols
Void.

3.3 Abbreviations
For the purposes of the present document, the abbreviations given in ETSI GS NFV 003 [i.2] and the following apply:

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA</td>
<td>Certificate Authority</td>
</tr>
<tr>
<td>CMS</td>
<td>Cryptographic Message Syntax</td>
</tr>
<tr>
<td>CSAR</td>
<td>Cloud Service ARchive</td>
</tr>
<tr>
<td>IANA</td>
<td>Internet Assigned Number Association</td>
</tr>
<tr>
<td>TOSCA</td>
<td>Topology and Orchestration Specification for Cloud Applications</td>
</tr>
<tr>
<td>URI</td>
<td>Universal Resource Identifier</td>
</tr>
<tr>
<td>UTF</td>
<td>Unicode Transformation Format</td>
</tr>
<tr>
<td>YAML</td>
<td>YAML Ain't Markup Language</td>
</tr>
</tbody>
</table>

4 NSD file structure

4.1 TOSCA YAML Cloud Service Archive (CSAR)

4.1.1 CSAR structure
A TOSCA YAML CSAR file is an archive file using the ZIP file format whose structure complies with the TOSCA Simple Profile in YAML version 1.2 specification [2]. The CSAR file may have one of the two following structures:

- CSAR containing a TOSCA-Metadata directory, which includes the TOSCA.meta metadata file providing an entry information for processing a CSAR file as defined in TOSCA v1.0 Specification [i.1].

- CSAR containing a single yaml (.yml or .yaml) file at the root of the archive. The yaml file is a TOSCA definition template that contains a metadata section with template_name and template_version metadata. This file is the CSAR Entry-Definitions file.

In addition, the CSAR file may optionally contain other directories with bespoke names and contents.
4.1.2 TOSCA Entry definition file metadata extension

4.1.2.1 Metadata keynames

Table 4.1.2.1-1 specifies an extension of the list of recognized metadata keynames as specified in TOSCA-Simple-Profile-YAML-v1.2 [2] for the main TOSCA Service Template.

<table>
<thead>
<tr>
<th>Keyname</th>
<th>Required</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>yang_definitions</td>
<td>No</td>
<td>string</td>
<td>Reference to a YANG file representing the NSD within an NSD file archive.</td>
</tr>
</tbody>
</table>

4.1.2.2 Additional requirement

If a YANG-based NSD is included in the NSD file archive, the main TOSCA definitions YAML file shall include a metadata section with a metadata entry, where the keyname is "yang_definitions" and the value is the path to the YANG file representing the NSD within the NSD file archive. No additional contents shall be included in the main TOSCA definitions YAML file.

EXAMPLE:

```
tosca_definitions_version: tosca_simple_yaml_1_2

metadata:
    template_name: MainServiceTemplate
    template_author: Onboarding portal
    template_version: 1.0
    yang_definitions: Definitions/myNSD.xml
```

END OF EXAMPLE.

4.2 NSD file structure and format

The structure and format of an NSD file archive shall conform to the TOSCA Simple Profile in YAML version 1.2 specification of the CSAR format [2].

NOTE: This implies that the NSD file archive can be structured according to any of the two options described in clause 4.1.

4.3 NSD file contents

4.3.1 General

An NSD file archive shall contain the NSD as a main TOSCA definitions YAML file, representing all or part of the NSD, and additional files. It shall be structured according to one of the CSAR structure options described in clause 4.1.

NOTE 1: ETSI GS NFV-SOL 001 [i.3] specifies the structure and format of the NSD based on TOSCA specifications.

NOTE 2: ETSI GS NFV-SOL 006 [i.4] specifies the structure and format of the NSD based on YANG specifications.
If the option with a TOSCA-Metadata directory is used and the CSAR-Version parameter indicates version 1.0, all files that are contained in the archive shall be referenced from the TOSCA.meta file. If the CSAR-Version parameter indicates version 1.1, the files that are referenced and pointed to by relative path names through artifact definitions in one of the TOSCA definitions files (e.g. the NSD) contained in the CSAR need not be declared in the TOSCA.meta file.

If a YANG-based NSD is included in the NSD file archive only the option without a TOSCA-Metadata directory is applicable.

Examples of NSD file archive options are described in annex A.

4.3.2 NSD file archive manifest file

A CSAR NSD file archive shall contain a manifest file. The manifest file shall have an extension .mf and the same name as the main TOSCA definitions YAML file and be located at the root of the archive (archive without TOSCA-Metadata directory) or in the location specified by the TOSCA.meta file (archive with a TOSCA-Metadata directory). In the latter case, the corresponding entry shall be named "Entry-Manifest".

The manifest file shall start with the NSD file archive metadata in the form of a name-value pairs. Each pair shall appear on a different line. The "name" and the "value" shall be separated by a colon and, optionally, one or more blanks. The name shall be one of those specified in table 4.3.2-1 and the values shall comply with the provisions specified in table 4.3.2-1.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>nsd_release_date_time</td>
<td>String formatted according to IETF RFC 3339 [3].</td>
</tr>
<tr>
<td>nsd_file_structure_version</td>
<td>A string. See note 2.</td>
</tr>
</tbody>
</table>

NOTE 1: The value shall be identical to that specified in the NSD.
NOTE 2: The value shall be identical to the version attribute specified in the NSD.

An example of valid manifest file metadata entries follows.

EXAMPLE 1:

```yaml
metadata:
nsd_designer: Mycompany
nsd_invariant_id: Sunshine
nsd_name: Sunshine
nsd_file_structure_version: 1.0
nsd_release_date_time: 2018-04-08T10:00+08:00

END OF EXAMPLE 1.
```

The manifest file shall include a list of all files contained in or referenced from the NSD file archive with their location, expressed using a Source: location/name key-value pair. The manifest file itself may be included in the list.

Below is an example of valid manifest file entries for files contained in or referenced from the NSD file archive.

EXAMPLE 2:

```yaml
Source: SunShine.yaml
Source: scripts/install.sh
Source: https://www.designer.org.com/SunShine/v4.1/scripts/scale/scale.sh

END OF EXAMPLE 2.
```
If the NSD file archive refers to external files, the manifest file shall contain digests of individual files in the file archive, both local files contained in the archive and external files referenced in the archive.

If the NSD file archive does not refer to external files, the manifest files may contain digests of the individual files contained in the archive. If the manifest file does not include digests, the complete CSAR file shall be digitally signed by the NS designer. A consumer of the NSD file archive verifies the digests in the manifest file by computing the actual digests and comparing them with the digests listed in the manifest file.

The manifest file, or alternatively, the signature of the CSAR file, is the key for decision regarding an NSD file archive integrity and validity in terms of its contained artifacts. The specification of the manifest file and specific algorithms used in digest creation and validation is described in the security related clause.

The details of specifying the local or externally located files and their security protection are described in clause 5.

### 4.3.3 NSD file archive change history file

A CSAR NSD file archive shall contain a humanly readable text file describing any change in the constituency of the NSD file archive. All the changes in the NSD file archive shall be versioned, tracked and inventoried in the change history file.

The NSD file archive change history file shall be named "ChangeLog.txt" and be located at the root of the archive (archive without TOSCA-Metadata directory) or in the location specified by the TOSCA.meta file (archive with a TOSCA-Metadata directory). In the latter case, the corresponding entry shall be named "Entry-Change-Log".

### 4.3.4 Testing files in the NSD file archive

To enable NS validation, an NS designer should include in an NSD file archive, files containing necessary information (e.g. test description) in order to perform NS testing. The contents of NS testing information included in the NSD file archive is outside the scope of the present document.

The NS testing information in the NSD file archive shall be located in a directory named "Tests" located at the root of the archive (archive without TOSCA-Metadata directory) or in the location specified by the TOSCA.meta file (archive with a TOSCA-Metadata directory). In the latter case, the corresponding entry shall be named "Entry-Tests".

### 4.3.5 Certificate file

If the manifest file is signed by the NS designer (see option 1 in clause 5.1), the CSAR NSD file archive shall contain a certificate file if the certificate is not included in the signature container (see note) within the manifest file. In this case or if a single certificate is provided for the signature of multiple artifacts (see clause 5.4), the certificate file shall have an extension .cert and the same name as the main TOSCA definitions YAML file and be located at the root of the archive (archive without TOSCA-Metadata directory) or in the location specified by the TOSCA.meta file (archive with a TOSCA-Metadata directory). In the latter case, the corresponding entry shall be named "Entry-Certificate".

**NOTE:** Signature container refers to a structure in a standard format (e.g. CMS) which contains signature and additional data needed to process the signature (e.g. certificates, algorithms, etc.).

If the complete CSAR file is signed by the NS designer (see option 2 in clause 5.1), the certificate file shall be contained in a zip file together with the CSAR file and the signature file if the certificate is not included in the signature file. The certificate file shall have an extension .cert and the same name as the CSAR file.

### 5 Adding security to TOSCA CSAR

#### 5.1 NSD file archive authenticity and integrity

An NSD file archive shall support a method for authenticity and integrity assurance.
In order to provide the public key based authenticity and integrity for the whole NSD file archive one of the two following options shall be followed:

Option 1: The NSD file archive shall contain a Digest (a.k.a. hash) for each of files it contains. The table of hashes shall be included in the manifest file, which is signed with the NS designer private key. In addition, the NS designer shall include a signing certificate that includes the NS designer public key, following a predefined naming convention and located either at the root of the archive or in a predefined location (e.g. directory).

The certificate may also be included in the signature container, if the signature format allows that. For example, the CMS format allows to include the certificate in the same container as the signature.

Option 2: The complete CSAR file shall be digitally signed with the NS designer private key. The NS designer delivers one zip file consisting of the CSAR file, a signature file and a certificate file that includes the NS designer public key. The certificate may also be included in the signature container, if the signature format allows that.

Option 2 is only valid if all artifacts are included in the NSD file archive, i.e. no external artifacts are referenced from the files contained in the NSD file archive.

This solution, either option 1 or option 2, relies on the existence in the NFVO of a root certificate of a trusted CA that shall have been delivered via a trusted channel that preserves its integrity (separate from the NSD file structure) to the NFVO and be preinstalled in the NFVO before the on-boarding of the NSD file structure.

NOTE: The present document makes no assumption on who this trusted CA is. Furthermore, it does not exclude that the root certificate be issued by the NS designer or by the NFVI provider.

5.2 Manifest and certificate files in the NSD file archive

In option 1 (see clause 5.1) the manifest file provides integrity assurance of the NSD file archive. In this option the manifest contains the digests (hashes) for each individual file locally stored within the NSD file archive or referenced from it. Each file related entry of the manifest file includes the path or URI of the individual file, the hash algorithm and the generated digest. A consumer of the NSD file archive shall verify the digests in the manifest file by computing the actual digests and comparing them with the digests listed in the manifest file.
In option 1 authenticity of the NSD file archive is ensured by signing the manifest file with the NS designer private key. The digital signature is stored in the manifest file itself (see clause 5.3). The NS designer shall include an X.509 certificate [4] in the NSD file archive. The certificate shall be either placed in a certificate file with extension .cert or, if the chosen signature format allows it, the certificate may be included in the signature container itself. The certificate provides the NS designer public key. In a CSAR file without metadata directory the .cert file shall have the same name as the TOSCA definitions YAML file and be located at the root of the archive (archive without TOSCA-Metadata directory). In a CSAR file with a metadata directory, the .cert file shall be placed or in the location specified by the TOSCA.meta file (archive with a TOSCA-Metadata directory). In the latter case, the corresponding entry shall be named "Entry-Certificate".

In option 2 (see clause 5.1), authenticity and integrity of the NSD file archive is ensured by signing the CSAR file with the NS designer private key (option 2 in clause 5.1). The digital signature is stored in a separate file. The NS designer shall also include an X.509 certificate in a separate file with extension .cert or, if the signature format allows it, in the signature file itself. The NS designer creates a zip file consisting of the CSAR file, signature and certificate files. The signature and certificate files shall be siblings of the CSAR file with extensions .cms and .cert respectively.

In this alternative (option 2 in clause 5.1) it is not required to include digests (hashes) per each individual file or artifact in the manifest file but it is recommended to include individual signatures of the artifacts (see clause 5.4). A consumer of the NSD file archive can verify the signature of the complete CSAR file with the NS designer public key.

Table 5.2-1 summarizes the characteristics of the two possible options for integrity assurance.

<table>
<thead>
<tr>
<th>Options</th>
<th>Digest per artifact</th>
<th>Signature per artifact</th>
<th>Support external artifacts</th>
<th>Signature as part of the manifest file</th>
<th>External Signature file for the whole CSAR</th>
<th>Certificate may be part of the signature</th>
<th>Certificate may be in a separate file</th>
</tr>
</thead>
<tbody>
<tr>
<td>Option 1</td>
<td>Yes</td>
<td>Yes (allowed)</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Option 2</td>
<td>No</td>
<td>Yes (recommended)</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

The X.509 certificate may contain one single signing certificate or a complete certificate chain. The root certificate that may be present in this X.509 certificate file shall not be used for validation purposes. Only trusted root certificate pre-installed in NFVO shall be used for validation (see clause 5.1).

5.3 Conventions in the manifest file

When the Manifest file provides the integrity assurance of the NSD file archive (option 1 in clause 5.1) it shall contain a list of blocks of name-value pairs, where each block is related to one file in the NSD file archive, where name and value are separated by a colon and, optionally, one or more blanks. Each block shall contain the following three name-value pair attributes:

- Source: identifier of the file used as input to the hash generation algorithm. The source can be either:
  - A file name for a file that is contained in the root of the CSAR archive.
  - A file name with path for a file in the CSAR archive that is not contained in the root of this archive.
  - A URI to an externally accessible artifact.

- Algorithm: name of a well-known algorithm used to generate the hash.

- Hash: text string corresponding to the hexadecimal representation of the hash.

The value for the Algorithm name-value pair shall be among those registered by IANA for hash function textual names [5]. An NSD file archive that complies with the present document shall either use "sha-256" or "sha-512".

Including the hash algorithm in each entry is optional if it is communicated by other means.
If option 1, as defined in clause 5.1, applies, the manifest file shall be signed. Otherwise signing the manifest file is optional. When the manifest file is signed, the signature shall be included at the end of the file. The signature and all necessary data to interpret it (algorithm used to generate the hash and encryption method) shall be included in an archive in a standard format following digital signatures best practices and encoded in a textual representation according to IETF RFC 7468 [6]. The format shall be among those registered by IANA for mime types [7] (e.g. "cms", "pkcs8", etc.).

Example of valid manifest file entries including manifest signature in CMS format.

**EXAMPLE:**

Source: SunShine.yaml
Algorithm: SHA-256
Hash: ead2ca54bfd94b72fb210e6b69d24f870f775673a271be7726f

Source: https://www.designer.org.com/SunShine/v4.1/scripts/install.sh
Algorithm: SHA-256
Hash: 16bb3cd7c2d685e0b6da9b1f3f67a11efba692d02d9870f775673a271be7726f

---BEGIN CMS-----
MIIFkwYJKoZIhvcNAQcCoIIFhDCCBYAcqATCAhBgUrDgMCGjcCACgsGzS1b3DQ8HACAzc2AxgQGUZam1wO18bZT5fW8syDfB8qBodvcm3Baa0sG1IbQ5S6pQXNwN2Q2Wz4h7YjTUYV3S5QnTROZyIyJ1wMOUXVycJMDqTW3yK7yjMdJ1D3CjN3B2QW0Q3FnmZ2WEYNm0xT1j2WzKyTycGucOU291cm1n01lBzyZy3JpRhzL21uc3hBbwu2gNCzFZ2r2yaXRo8cbgoU09bBLTI1Ng0KSGFZFaZaogMTI2YjYjNZjDdMjM9Q2DVM1Z2B3G5yYP56mFmzMYZ2N8x8MWvYmE2OTJX0DcmNhjyjMhMhMhMhMhMjxMnJ3NnjMjz2s3KRoQZ2B3VYyU26I9h0dHBo2Lon8v33dL3L31c2l3nbnVyX29yYyYjZjbo2U3VUV2hpmVldjQX89YeYjJpCRZL3JvYXlLL2NcDQpBb8d4dcm1oAg66FNIQ5Qy5TNYtNcKhCG2g6D1k02mVkJ2AyjYWYwzdmoGQ00c0ZjJyaNdkdODU1NwV
MTY3YyQ4ZTM2MjH1y3X3OTpHTM1jYdkdWNWNGZ12WggqIPIMIICCcC2Acq AwIBAglJAKGuG1cNcMw9KeMANcGGCM49BAMCMCMeC11A8JgAvNBAYTAmEuOqCQYDVQQI
DAJqgELMMcSAL225Z65CmnomD0AKBvGNaAMh3p3O2TM00NaGfunctionCw0uG1mMB4X
DTE4MDNw0TA2NTQyVx5DXTI4MDMw0J2A3NpyVx0QvGe5YiELNk0AUEBhMCY24czCAJ
BnVBAgmpqM6wQx0CQVQHQCADbn1MAkDEQDEQcCGBYq2WHQYDQQLQESTMD
bWwWd1AQBqcgchkBQPIB8BqurqQA1qG1NLAASJdQxszB3Jiid51WmqwRt2r12f9YqVnVr
mvUQmv/abcd:0bJF7V2y2bXW3mHs1B1eqGQW1Y0kQPLnzK0pxP1tjTR0sQ8pt1Y
5YX11b7tP8sGD6R5J58QhrnNnQVo1Dyre3jUDB0MBG0A1udDqQWDBBRS/0ece0x
mB0JUE4RbXxJAFbgDNVHMSGEBdWxQ/0ece0xsm8wT3quHEKrBrxXKwJAM
BqNHVHREBTADQA/MAC0GCcGM49BAMC2akAMyGMCYCQ49eF/q7/17seVE7BMI+BS678
DYFPsSRQ23CornyX9TEUeY11Gh2y3yHtyZd4D2QrrjwCMQدولIINlHeBzDJ6hSdV
3Q1c8EgyUDlYvTrqEpy40QxRvXZV21zEr7wI1h0Fdx9ggwGmW1IBaIA8BTBQ
MN3QeZYgZBvNBAYTAM3eQnWxQYDVQVDCjACzCzELMkGeIUEBwCboexxD0ARKwB
BAAcMA3p0ZTEMBAGCAEwDsaGlA1mgA0YaWdmgxuQWw6Fyp4D4hggd1GAYJ
KoZ1hvcnaQbMDQsGCGSisG3BQHATAcBqgkhh1G9w0BCQbXdxoNTFwO71A6DMD
NTM9jA8jkh1g9k1h9w0BCQbXqFqUSM/XnCc0zTuUY2z21117ThY30UweQ1YKQoZ1
hvcNAQk3M0WwaaJABgHkXhBGEA2SxwVj1y1iADBXABE4BAMGCG0SFA1wqB
AJfBBZggQQgB2eQzB2Q2bGggQgB16hW0AJS7ccAIb0QYIKoZIhvcNCAwCJAUxWBY7F
W4dAgg1QMyZIhvcIAICcAgIQYIHoZ1jI0EAOEJQWMMbQCMCM7CZcw46jubJHd
q6G5Qf1vMMB0V0ExDSDoJ9WzeGZhbc1056knB0P/uzxw7VY2A1WxU/2121nhAd
/9UQGKQ6z3W7W/8t8MbMRZSTETanOF0FDrFPf9unsvkcV9Q0ReZ

---END CMS-----

### 5.4 Signature of individual artifacts

The NS designer may optionally digitally sign some or all artifacts individually. This option exists for both option 1 and option 2 described in clause 5.1 but it is recommended when no individual hashes per artifact are included (i.e. in option 2 in clause 5.1). In this case a signature file in standard format (e.g. CMS, PKCS#7) should be included in the archive. The signature shall follow the same name (different extension) as the signed artifact and be a sibling of it, i.e. placed in the same folder in the archive, which could also be the root of the archive.
A certificate shall also be included in the NSD file archive as per one of the two following alternatives:

- One certificate per signed artifact:
  Either a certificate file with extension .cert is included as a sibling of the signed artifact file, i.e. placed in the same folder as the signed artifact and having the same name (different extension) or the certificate is included in the signature file, provided that the signature format allows for it. This alternative allows to have different certificates per different artifacts, which may be needed e.g. if artifacts contained in the NSD file archive are signed by 3-rd party designers.

- One single certificate for all signed artifacts:
  One certificate file with extension .cert and the same name as the main TOSCA definitions YAML file and located at the root of the archive (archive without TOSCA-Metadata directory) or in the location specified by the TOSCA.meta file (archive with a TOSCA-Metadata directory). In the latter case, the corresponding entry shall be named “Entry-Certificate”.

5.5 Support for security sensitive artifacts

If an artifact is security sensitive, the whole artifact may be encrypted by the NS designer with an artifact specific key. In case of asymmetric encryption this key is a public key provided by the party who is responsible to on-board and validate the NSD file archive or to use the artifact, and the NS designer uses it to encrypt the security sensitive artifact. The consumer of this artifact then decrypts the artifact with its own private key.

In case of symmetric encryption, the public key provided by the party responsible to on-board and validate the NSD file archive or to use the artifact is used to encrypt a key generated by the NS designer. The artifact is encrypted with this latter key, which is to be shared with the consumer of the artifact and shall be included in encrypted form in the NSD file archive. The consumer of the artifact decrypts the shared key with its own private key and then uses the obtained shared key to decrypt the artifact.

In this scenario the encrypted artifact shall be delivered in a CMS file [8], which provides all necessary information to decrypt it: algorithm used for the artifact encryption, encrypted key used for artifact encryption and algorithm used to encrypt the key.

The encryption of an artifact occurs prior to the generation of a digest (hash) for the artifact.
Annex A (informative):
TOSCA CSAR Examples

A.1 CSAR with the TOSCA-Metadata directory

Below is an example of a CSAR directory archive including the TOSCA-Metadata, Definitions, Files and Scripts directories. The TOSCA-Metadata directory contains the TOSCA.meta file as specified in [i.1]. The NSD (Sunshine.yaml) and other templates files, if any, are included in the Definitions directory. The Files directory contains the change log file, certificate file and other artifact files. The Scripts directory includes the scripts files that may be called from the NSD. The manifest file (Sunshine.mf) is located at the root level of the archive.

EXAMPLE:

!------ TOSCA-Metadata
    !------ TOSCA.meta

!------ Definitions
    !------ Sunshine.yaml
    !------ Other Templates (e.g., type definitions)

!------ Files
    !------ ChangeLog.txt
    !------ Sunshine.cert
    !------ Other artifacts
    !------ Tests
      !------ file(s)
    !------ Licenses
      !------ file(s)

!------ Scripts
    !------ install.sh

!------ Sunshine.mf

END OF EXAMPLE.

A.2 CSAR without the TOSCA-Metadata directory

Below is the example of a CSAR directory structure including the NSD (Sunshine.yaml), manifest, certificate, testing, licensing and change log files located at the root level of the CSAR. The Artifacts directory includes the two scripts files that may be called from the NSD.

EXAMPLE:

!-------- Sunshine.yaml
!-------- Sunshine.mf
!-------- Sunshine.cert
!-------- ChangeLog.txt

!-------- Tests
    !------ file(s)

!-------- Licenses
    !------ file(s)

!-------- Artifacts
    !------ install.sh
    !------ start.yang

END OF EXAMPLE.
A.3 CSAR with the YANG NSD without TOSCA.meta directory

Below is an example of CSAR including the NSD (CompanyNSD.xml), a main TOSCA definition YAML file with metadata only (CompanyNSD.yaml), manifest, certificate, licensing and change log files located at the root level of the CSAR. The Scripts directory includes one script file that may be called from the NSD. This example does not preclude having other YAML files at other locations than the root of the CSAR file.

EXAMPLE:

```bash
!——— CompanyNSD.yaml
!——— CompanyNSD.xml
!——— CompanyNSD.mf
!——— CompanyNSD.cert
!——— ChangeLog.txt

!——— Files

!——— Instance Data Files

!——— start.xml

!——— Licenses

!——— Scripts

!——— install.sh

END OF EXAMPLE.
```
Annex B (informative):
Authors & contributors

The following people have contributed to the present document:

**Rapporteur:**
Manchang Ju, ZTE

**Other contributors:**
Ernest Bayha, Ericsson
Andrei Kojukhov, Amdocs
Bhumip Khasnabish, ZTE
Dmytro Gassanov, Netcracker
Jie Zhan, ZTE
Shitao Li, Huawei
Maopeng Zhang, ZTE
Baoguo Xie, ZTE
Bruno Chatras, Orange
Mahesh Jethanandani, VMware
Annex C (informative):

Change History

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Information about changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>April 2018</td>
<td>0.0.1</td>
<td>Initial version based on contributions that were agreed at the NFVSOL#57 meeting and during Email Approval (EA) following the NFVSOL#57.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000105_ETSI_GS_NFV_SOL007_ToC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000106_ETSI_GS_NFV_SOL007_Scope</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000107_ETSI_GS_NFV_SOL007_Normative_References</td>
</tr>
<tr>
<td>April 2018</td>
<td>0.0.2</td>
<td>Version 0.0.2 based on contributions that were agreed at the NFVSOL#60 meeting and during Email Approval (EA) following the NFVSOL#59 and NFVSOL#60 meetings.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000140_SOL007_Adding_Normative_Reference</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000141_SOL007_Normative_text_for_CSAR_Structure_options</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000149_SOL007_Adding_4_1_TOSCA_YAML_CSAR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000150_SOL007_Adding_Informative_References</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000151_SOL007_Clause_3_-_Fixing_Definitions_and_Abbreviations</td>
</tr>
<tr>
<td>August 2018</td>
<td>0.0.3</td>
<td>Version 0.0.3 based on contributions that were agreed at the NFVSOL#74 meeting and during Email Approval (EA) following the NFVSOL#61 and NFVSOL#74 meetings.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000163_SOL007_Adding_NSD_file_contents_General</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000468r3_SOL007_NSD_file_structure_manifest_metadata</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000476r2_SOL007_Appendix_A_CSAR_example_with_the_TOSCA_Metadata_director</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000480_SOL007_NSD_file_structure_change_history_file</td>
</tr>
<tr>
<td>September 2018</td>
<td>0.1.0</td>
<td>Version 0.1.0 based on contributions that were agreed at the NFVSOL#75 and NFVSOL#77 meetings.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000484r2_SOL007_NSD_file_structure_testing_files</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000509_SOL007_Adding_security_to_TOSCA_CSAR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000512_SOL007_NSD_file_structure_authenticity and integrity_option2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000513_SOL007_Support_for_security_sensitive_artifacts</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000527_SOL007_Conventions_in_the_manifest_file</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000531_SOL007_Signature_of_individual_artifacts</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000532_SOL007_Certificate_file</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000533_SOL007_Clause_5_2_manifest_and_certificate_files</td>
</tr>
<tr>
<td>October 2018</td>
<td>0.2.0</td>
<td>Version 0.2.0 based on contributions that were agreed at the NFVSOL#81 meeting.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)0005568_SOL007_Appendix_A_CSAR_example_without_the_TOSCA_Metadata_director</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000597_SOL007_-_Various_Wording_Improvements</td>
</tr>
<tr>
<td>January 2019</td>
<td>2.5.2</td>
<td>Version 2.5.2 based on contributions that were agreed during Email Approval (EA) following the NFVSOL#86 meeting.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000758r1_SOL007_Adding_support_for_YANG_NSD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(18)000769_SOL007_Adding_Informative_references_for_YANG_NSD</td>
</tr>
<tr>
<td>January 2019</td>
<td>2.5.3</td>
<td>Version 2.5.3 based on contributions that were agreed during Email Approval (EA) following the NFVSOL#87 and the NFVSOL#88 meeting.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(19)000007_SOL007_clarifying_the_using_of_a_list_of_artifacts_in_the_ma</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(19)000006_SOL007_correcting_a_date_time_format</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(19)000005r1_SOL007_An_improvement_in_clause_4_3_2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- NFVSOL(19)000027_SOL007_An_improvement_for_TOSCA_YAML_version_in_clause_4_1_2</td>
</tr>
</tbody>
</table>
### History

<table>
<thead>
<tr>
<th>Document history</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>V2.5.1</strong></td>
</tr>
<tr>
<td><strong>V2.6.1</strong></td>
</tr>
</tbody>
</table>