DRAFT

GUIDE TO OSCAL-BASED FEDRAMP CONTENT

Version 1.0

July 1, 2020

FedRAMP

DOCUMENT REVISION HISTORY

Date	Description	Version	Author
7/1/2020	Initial Publication	1.0	FedRAMP PMO
<Date>	<Revision Description>	<Version>	<Author>
<Date>	<Revision Description>	<Version>	<Author>

How to Contact Us

For questions about FedRAMP, or for technical questions about this document including how to use it, contact *info@FedRAMP.gov.*

For more information about FedRAMP, see https://FedRAMP.gov.

TABLE OF CONTENTS

1. OVERVIEW

1.1. Who Should Use This Document?

This document is intended for technical staff and tool developers implementing solutions for importing, exporting, and manipulating Open Security Controls Assessment Language (OSCAL)-based FedRAMP content, such as system security plans (SSP), security assessment plans (SAP), security assessment reports (SAR), and plans of action and milestones (POA&M).

It provides guidance and examples for an organization producing and using OSCAL-based, FedRAMP-compliant files. Our goal is to enable your organization to develop tools that will seamlessly ensure these standards are met so your security practitioners can focus on content and accuracy rather than formatting and presentation.

1.2. Related Documents

This document does not stand alone. It provides foundational information and core concepts, which apply to the following four guides:

- *Guide to OSCAL-based FedRAMP System Security Plans (SSP)*
- *Guide to OSCAL-based FedRAMP Security Assessment Plans (SAP)*
- *Guide to OSCAL-based FedRAMP Security Assessment Reports (SAR)*
- *Guide to OSCAL-based FedRAMP Plan of Action and Milestones (POA&M)*

Each of those documents will refer back to this one for common concepts.

1.3. Basic Terminology

XML and JSON use different terminology. Instead of repeatedly clarifying format-specific terminology, we use the following format-agnostic terminology through these documents.

TERM	XML EQUIVALENT	JSON EQUIVALENT
`Field`	A single element or node that can hold a value or an attribute	A single object that can hold a value or property
`Flag`	Attribute	Property
`Assembly`	A collection of elements or nodes. Typically, a parent node with one or more child nodes.	A collection of objects. Typically, a parent object with one or more child objects.

These terms are used by the National Institute of Standards and Technology (NIST) in the creation of OSCAL syntax.

Throughout these documents, the following words are used to differentiate between requirements, recommendations, and options.

TERM	MEANING
must	Indicates a requirement.
should	Indicates a recommendation that is not necessarily required.
may	Indicates an available optional that may be used at the tool developer's discretion.

1.4. XML and JSON Formats

The examples provided here are in XML; however, FedRAMP accepts XML or JSON formatted OSCAL content. NIST offers the ability to convert OSCAL-files between XML and JSON in either direction without data loss.

You may submit your SSP, SAP, SAR, and POA&M to FedRAMP using either XML or JSON. If necessary, FedRAMP's tools will convert the files for processing.

> *For more information on converting OSCAL files between XML and JSON, see Section 1.6.2, NIST OSCAL Format Conversion Mechanisms.*

NOTE: NIST partially supports YAML Ain't Markup Language (YAML) as an offshoot of JSON. FedRAMP will evaluate the use of YAML for FedRAMP deliverables once NIST offers the same level of support for YAML syntax validation and format conversion as compared to XML and JSON.

1.5. OSCAL-based FedRAMP Templates

FedRAMP offers OSCAL-based templates in both XML and JSON formats for the SSP, SAP, SAR, and POA&M. These templates contain many of the FedRAMP required content and placeholders to help get you started. This document is intended to work in concert with those templates. The OSCAL-based FedRAMP templates are available here:

- https://github.com/GSA/fedramp-automation/raw/master/templates

1.6. XML and JSON Technology Standards

For OSCAL compliance, mechanisms that interpret or generate OSCAL content must honor the core syntax described at https://pages.nist.gov/OSCAL/documentation/schema/.

While not mandatory, organizations adopting OSCAL are strongly encouraged to use the NIST-published validation and translation mechanisms. The validation mechanism ensures XML and JSON files are using OSCAL-compliant syntax, while the translation mechanism converts OSCAL content from either format to the other. NIST has an automated governance process, which ensures these mechanisms remain aligned with the latest OSCAL syntax.

> ***TIP**: There are comments in the XML versions of the FedRAMP Templates. Unfortunately, JSON does not formally support comments. JSON users may wish to review the comments in the equivalent sections of the XML files.*

1.6.1. NIST OSCAL Syntax Validation Mechanisms

The latest version of NIST OSCAL schema validation files are always available here:

XML: https://github.com/usnistgov/OSCAL/tree/master/**xml**/schema
JSON: https://github.com/usnistgov/OSCAL/tree/master/**json**/schema

Validating XML-based OSCAL files using the NIST-published schema validation requires:

XML Schema Definition Language (XSD) 1.1
[https://www.w3.org/TR/xmlschema11-1/]

Validating JSON-based OSCAL files using the NIST-published schema validation requires:

JSON Schema, draft-07
[https://json-schema.org/]

There are several open-source and commercial tools that will process XSD 1.1 or JSON Schema, draft-07, either as stand-alone capabilities or as programming libraries.
FedRAMP and NIST are unable to endorse specific products.

1.6.2. NIST OSCAL Format Conversion Mechanisms

The latest version of NIST OSCAL format conversion files are always available here:

XML to JSON: https://github.com/usnistgov/OSCAL/tree/master/**json**/convert
JSON to XML: https://github.com/usnistgov/OSCAL/tree/master/**xml**/convert

Converting between XML and JSON in either direction using the NIST built the XML conversion files requires:

Extensible Stylesheet Language Transformation (XSLT) 3.0
[https://www.w3.org/TR/2017/REC-xslt-30-20170608/]

and:

XPath 3.1
[https://www.w3.org/TR/xpath-31/]

There are several open-source and commercial tools that will process XSLT3.0 and XPath 3.1, either as stand-alone capabilities or as programming libraries.
FedRAMP and NIST are unable to endorse specific products.

1.7. XPath Queries and References

XPath is a standard query language for XML files, and is available natively in most modern programming languages. Even if you do not use XPath to query OSCAL data files, the XPath queries provide a concise and non-ambiguous way to communicate where the data is located within the file.

Except where noted, all XPath queries in this document are based on XPath 2.0. Most modern programming languages make XPath 1.0 available by default. XPath 2.0 can typically be added with third-party libraries, or calls to external command-line utilities.

> ***JSON Users**: There are several JSON query technologies available, such as JSONPath [https://restfulapi.net/json-jsonpath/]; however, no one technology has emerged as a clear standard as of this publication.*

The XPath queries in these guides have been designed to easily convert from XPath 2.0 to 1.0 using a function similar to the one on the next page, which enables developers to use native XML capabilities.

Sample PHP Code to Prepend Namespace (Converts *most* XPath 2.0 to 1.0)

```
// $query is the XPath 2.0 query
// $ns is the namespace value which much be prepended to element
//      names in the query
// RETURNS: An XPath 1.0 query
function AddNamespace2xpath($query, $ns) {
      $result = "";
      $q_len = strlen($query);
      $prev_char = "";

      for($i=0; $i < $q_len ; $i++) {
            $cur_char = substr($query, $i, 1);
            if ($prev_char === '/') {
                  if (ctype_alpha($cur_char)) {
                        $result .= $ns . ":";
                  };
            }
            $result .= $cur_char;
            $prev_char = $cur_char;
      }

      return $result;
}
```

Most XPath queries in this document are absolute paths from the root of the document. In other words, it is clear from the XPath query which of the major file sections described in Section 2.3 is being referenced, and where the field is located within the section.

***Database Users**: Some tool developers prefer to manage OSCAL data by first importing it into a database, which is a perfectly acceptable approach. Any OSCAL file generated form the database must still follow the OSCAL syntax. If the file is intended for delivery to FedRAMP, it must also follow the guidance in these guides.*

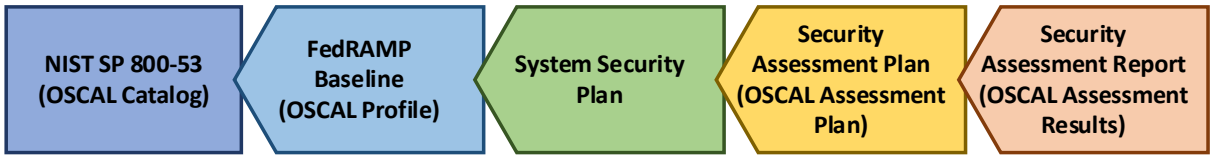
There are also database-like XML servers, such as the open-source tool BaseX [http://www.basex.org/], which allow OSCAL files to remain in their native format, yet be queried more like a traditional database. These XML databases typically optimize queries for better performance. Some will handle both XML and JSON files.

2. WORKING WITH OSCAL FILES

This section covers several important concepts and details that apply to OSCAL-based FedRAMP files.

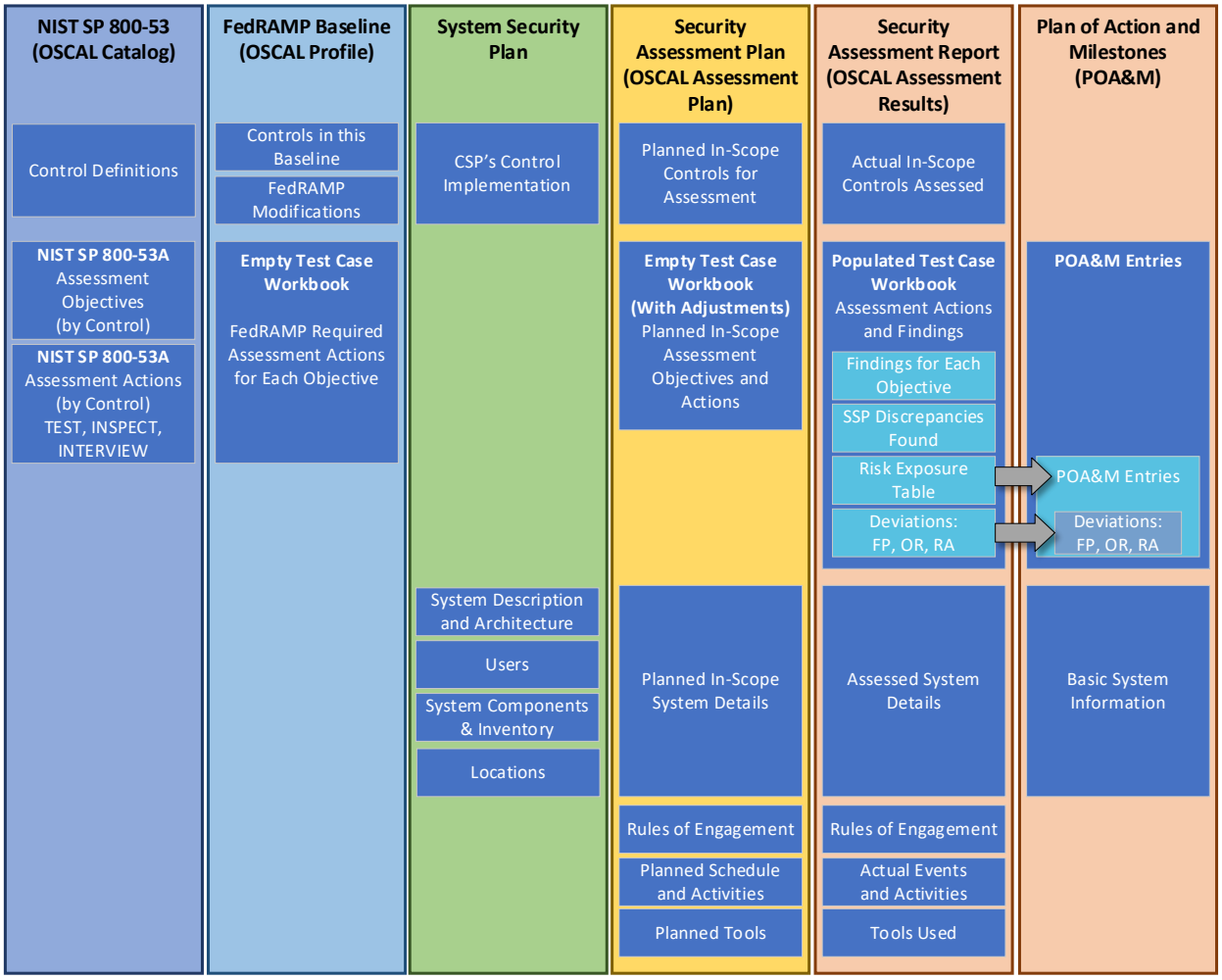
2.1. File Content Concepts

Unlike the traditional MS Word-based SSP, SAP, and SAR, the OSCAL-based versions of these files are designed to make information available through linkages, rather than duplicating information. In OSCAL, these linkages are established through `import` commands.



Each OSCAL file imports information from the one before it

For example, the assessment objectives and actions that appear in a blank test case workbook (TCW), are defined in the FedRAMP profile, and simply referenced by the SAP and SAR. Only deviations from the TCW are captured in the SAP or SAR.



Baseline and SSP Information is referenced instead of duplicated.

For this reason, an OSCAL-based SAP points to the OSCAL-based SSP of the system being assessed. Instead of duplicating system details, the OSCAL-based SAP simply points to the SSP content for information such as system description, boundary, users, locations, and inventory items.

The SAP also inherits the SSP's pointer to the appropriate OSCAL-based FedRAMP Baseline. Through that linkage, the SAP references the assessment objectives and actions typically identified in the FedRAMP TCW.

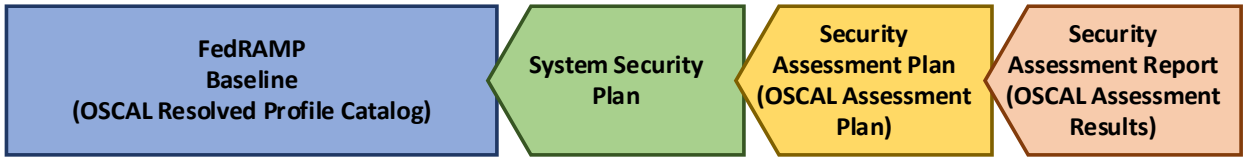
The only reason to include this content in the SAP is when the assessor documents a deviation from the SSP, Baseline, or TCW.

2.1.1. Resolved Profile Catalogs

The resolved profile catalog for each FedRAMP baseline is a pre-processing the profile and catalog to produce the resulting data. This reduces overhead for tools by eliminating the need to open and follow references from the profile to the catalog. It also includes only the catalog information relevant to the baseline, reducing the overhead of opening a larger catalog.

Where available, tool developers have the option of following the links from the profile to the catalog as described above, or using the resolved profile catalog.

Developers should be aware that at this time catalogs and profiles remain relatively static. As OSCAL gains wider adoption, there is a risk that profiles and catalogs will become more dynamic, and a resolved profile catalog becomes more likely to be out of date. Early adopters may wish to start with the resolved profile catalog now, and plan to add functionality later for the separate profile and catalog handling later in their product roadmap.



The Resolved Profile Catalog for each FedRAMP Baseline reduce tool processing

For more information about resolved profile catalogs, see *Appendix C, Profile Resolution*.

2.2. Hierarchy and Sequence

For both XML and JSON, the hierarchy of syntax is important and must be followed. For example, the `metadata` assembly must be at the top level of the OSCAL file, just below its root. If it appears within any other assembly, it is invalid.

The same field name is interpreted differently in different assemblies. For example, the `title` field under metadata is the document title, while the `title` field under role gives a human-friendly name to that role.

For XML, the sequence of fields and assemblies (elements) is also important. JSON typically does not require a specific sequence fields and assemblies (objects). Where sequence is important, OSCAL uses array objects in JSON.

For example, within the `metadata` assembly, XML must find `title`, `published`, `last-modified`, `version`, and `oscal-version` in exactly that order. The published field is optional and may be omitted, but if present, it must be in that position relative to the other fields. When using JSON, these fields are allowed in any order within the `metadata` assembly.

Tools must ensure this sequence is maintained when creating or modifying XML-based OSCAL files. NIST's XML documentation presents the fields and assemblies in the correct sequence.

Always use the hierarchy found here:

- SSP: https://pages.nist.gov/OSCAL/documentation/schema/ssp/xml-model-map/
- SAP: https://pages.nist.gov/OSCAL/documentation/schema/assessment-plan/xml-model-map/
- SAR: https://pages.nist.gov/OSCAL/documentation/schema/assessment-results/xml-model-map/
- POA&M: https://pages.nist.gov/OSCAL/documentation/schema/poam/xml-model-map/

2.2.1. Typical OSCAL Assembly Structure

Most assemblies in OSCAL follow a general pattern as follows:

- `title` (field): Typically only one `title` is allowed. Sometimes it is optional. This is a *markup-line* datatype.
- `description` (field): Typically only one `description` field is allowed. Sometimes it is optional. This is a *markup-multiline* datatype.
- `prop` (fields): There may be many `prop` fields. The property's `name` flag identifies the specific property. This is a *string* datatype.
- `annotation` (assemblies): There may be many `annotation` assemblies. The `name` flag identifies the specific annotation. The `value` flag is a string datatype and holds the intended value. The annotation assembly includes an optional `remarks` field, related to the defined name and value. The `remarks` field is a *markup-multiline* datatype.
- ***assembly-specific fields***
- `remarks` (field): Typically only one `remarks` field is allowed. It is always optional. This is a *markup-multiline* datatype.

While this is not universal in OSCAL, when any of these fields are present, they follow this pattern.

The `prop` fields and `annotation` assemblies are present to allow OSCAL extensions within each assembly. See *Section 3, FedRAMP Extensions, Conformity Tags, Defined Identifiers, and Accepted Values* for more information on FedRAMP's use of OSCAL extensions and FedRAMP conformity tags.

2.3. Multiple Layers of Validation

There are several layers at which an OSCAL file can be considered valid.
OSCAL-based FedRAMP files must be valid at all layers.

LAYER	DESCRIPTION
Well-Formed	The XML or JSON file follows the rules defined for that format. Any tool that processes the format will recognize it as "well-formed," which means the tool can proceed with processing the XML or JSON. XML: https://www.w3.org/TR/REC-xml/ JSON: https://json.org/
OSCAL Syntax	The XML or JSON file only uses names and values defined by NIST for OSCAL. NIST publishes schema validation tools to verify syntax compliance based on the following standards: XML Syntax Validation: XML Schema Definition Language (XSD) 1.1 JSON Syntax Validation: JSON Schema, draft 07
OSCAL Content	For certain OSCAL fields, the NIST syntax validation tools also enforce content - allowing only a pre-defined set of values to be used in certain fields. For example, Within the SSP model, impact levels within the information type assemblies only allow the following values: `fips-199-low`, `fips-199-moderate`, and `fips-199-high`. Any other value will cause an error when validating the file. In the future, NIST intends to publish more robust content enforcement mechanisms, such as Schematron. This enables more complex rules such as, "If field A is marked 'true', field B must have a value." FedRAMP is also exploring the use of Schematron for enhanced validation, and may publish these in the future.
FedRAMP Syntax Extensions	NIST designed OSCAL to represents the commonality of most cybersecurity frameworks and provided the ability to extend the language for framework-specific needs. FedRAMP makes use of these extensions. NIST provides `prop` and `annotation` fields throughout most of its assemblies, always with a `name`, `class`, and `ns` (namespace) flag: `<prop name="" class="" ns="">Data</prop>` In the core OSCAL syntax, the `ns` flag is never used. Where FedRAMP extends OSCAL, the value for `ns` is always: `https://fedramp.gov/ns/oscal` (case sensitive). When `ns='https://fedramp.gov/ns/oscal'` the `name` flag is as defined by FedRAMP. If the `class` flag is present, that is also defined by FedRAMP.
FedRAMP Content	Today, FedRAMP content is enforced programmatically. FedRAMP will evaluate the use of Schematron for future FedRAMP validation efforts.

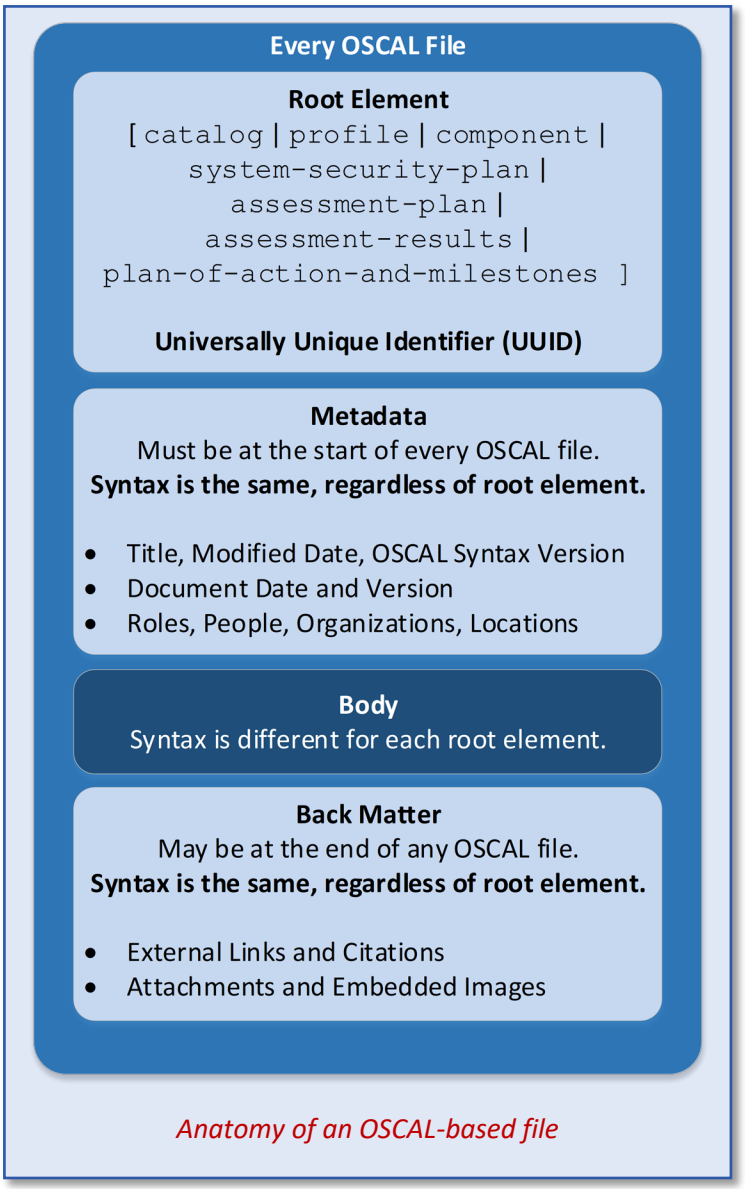
2.4. OSCAL's Minimum File Requirements

Every OSCAL-based FedRAMP file must have a minimum set of required fields/assemblies, and must follow the core OSCAL syntax found here:

https://pages.nist.gov/OSCAL/documentation/schema/

In addition to the core OSCAL syntax, the following FedRAMP-specific implementation applies:

- The root element of the file indicates the type of content within the body of the file. The recognized OSCAL root element types are as follows:
  - **catalog**: Contains a catalog of control definitions, as well as related assessment goals and activities, such as NIST SP 800-53 and 800-53A.
  - **profile**: Contains a control baseline, such as FedRAMP Moderate
  - **component**: Contains information about a product, service, or other security capability, such as the controls it can satisfy and what settings to use to meet the requirements of a particular baseline.
  - **system-security-plan**: Describes a system and its security capabilities, including its authorization boundary and a description of how each control is satisfied.
  - **assessment-plan**: Describes the plan for assessing a specific system.
  - **assessment-results**: Describes the actual activities performed in the assessment of a specific system, as well as the results of those activities.
  - **plan-of-action-and-milestones**: Describes and tracks known vulnerabilities of a system, as well as the plan for remediation.



Anatomy of an OSCAL-based file

- The Universally Unique ID (UUID) at the root must be changed every time the content of the file is modified.
- Every OSCAL file must have a metadata section and include a title, last-modified date, and OSCAL syntax version.

The table below shows an empty OSCAL file, based purely on the NIST syntax; however, FedRAMP requires much more in a minimum file. The latest OSCAL-based FedRAMP template files can be found here in JSON and XML formats:

https://github.com/GSA/fedramp-automation/tree/master/templates

An Empty OSCAL File Representation

```
<?xml version="1.0" encoding="UTF-8"?>
<OSCAL-root-element xmlns="http://csrc.nist.gov/ns/oscal/1.0"
   id="[generated-uuid]">
   <metadata>
      <title>Document Title</title>
      <last-modified>2019-11-27T00:00:00.00-05:00Z</last-modified>
      <version>0.0</version>
      <oscal-version>1.0-Milestone3</oscal-version>
   </metadata>

   <!-- body cut -->

   <back-matter />
</OSCAL-root-element>
```

2.4.1. UTF-8 Character Encoding

OSCAL uses UTF-8 character encoding. JSON files are always UTF-8 character encoded.

In XML, the first line in the example above ensures UTF-8 encoding is used. Other encoding options will create unpredictable results.

2.4.2. OSCAL Syntax Version

Tools designed to read an OSCAL file must verify the `oscal-version` field to determine which published syntax is used.

Tools designed to create or manipulate an OSCAL file must specify the syntax version of OSCAL used in the file in the `oscal-version` field.

NIST ensures backward compatibility of syntax where practical; however, this is not always possible. Some syntax changes between milestone releases leading up to OSCAL version 1.0 are unavoidable. NIST intends to keep all formally published schema validation files available, which keeps validation and conversion tools available for older versions of OSCAL. See Section *2.4.6 OSCAL Syntax Versions* for more information.

2.4.3. Content Change Requirements

Any time a tool changes the contents of an OSCAL file, it must also:

- update the file's `uuid` flag (`/*/@uuid`) with a new UUID; and
- update the `last-modified` field (`/*/metadata/last-modified`) with the current date and time. (Using the OSCAL date/time format, as described in *Section 2.6.1 Date and Time in OSCAL Files*)

Tools that open or import OSCAL files should rely on the UUID value provided by the `uuid` flag, and `last modified` field as easy methods of knowing the file has changed.

2.4.4. Cryptographic Integrity (Future)

NIST intends to add a cryptographic hash feature to OSCAL during calendar year 2020. Once added by NIST, FedRAMP will update this section.

While tool developers are encouraged to perform their own integrity checking, it is important to note cryptographic hash algorithms will produce a different result for inconsequential file differences, such as different indentation or a change in the sequence of flags.

2.4.5. Useful XPath Queries for Document Changes and OSCAL Syntax

Below are a few important queries, which enable a tool to obtain critical information about any OSCAL file.

XPath Queries

```
OSCAL syntax version used in this file:
  /*/metadata/oscal-version

Last Modified Date/Time:
  /*/metadata/last-modified

Unique Document ID:
  /*/@uuid

Document Title:
  /*/metadata/title
```

2.4.6. OSCAL Syntax Versions

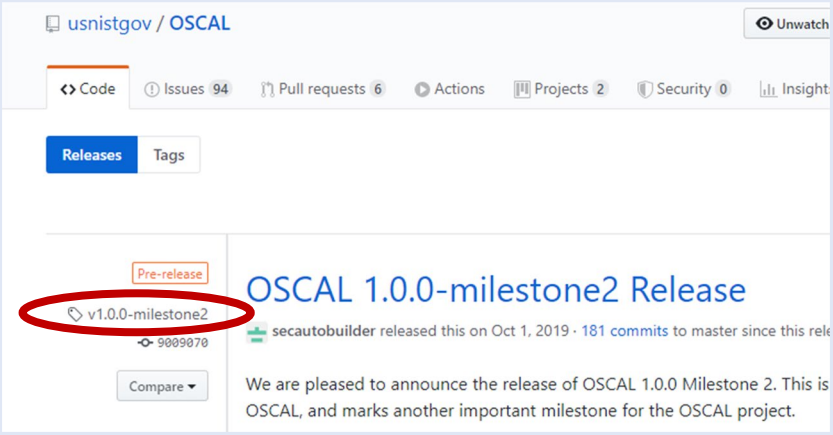
NIST's approach to OSCAL development ensures the syntax validation and format conversion tools remain available for each milestone and full release of OSCAL. As of this publication, the following syntax versions and associated resources are or will be made available:

SYNTAX VERSION	RESOURCES
Version 1.0.0 Full Release Identifier: `1.0.0`	*Expected September 2020*
Version 1.0.0 Pre-Release Milestone #3 Identifier: `1.0.0-milestone3`	*Expected June 2020* XML Schema Validation JSON Schema Validation XML to JSON Conversion JSON to XML Conversion
Version 1.0.0 Pre-Release Milestone #2 Identifier: `1.0.0-milestone2`	**Published** October 1, 2019 XML Schema Validation JSON Schema Validation XML to JSON Conversion JSON to XML Conversion
Version 1.0.0 Pre-Release Milestone #1 Identifier: `1.0.0-milestone1`	**Published** June 15, 2019 XML Schema Validation JSON Schema Validation XML to JSON Conversion JSON to XML Conversion

NIST always makes the latest version of syntax validation and format conversion files available in the master OSCAL repository, including any changes since the last formal release.

To ensure stable resources, use a formal OSCAL release. NIST publishes formal OSCAL releases here: https://github.com/usnistgov/OSCAL/releases

To access OSCAL resources based on a particular release, click the tag to the left of the release title.



2.5. Assigning Identifiers

NIST is introducing UUID flags with their OSCAL Milestone 3 release. Tools must generate UUIDv4 values and assign them anyplace a UUID flag exists.

There are three ways identifiers are typically used in OSCAL:

- **ID flag**: uniquely identifies a field or assembly.
- **UUID flag**: An RFC-4122 compliant universally unique identifier, version 4.
- **ID/UUID Reference**: a flag or field that points to another field or assembly using its unique ID or UUID flag value.
- **Uniform Resource Identifier (URI) Fragment**: A value in a flag or field with a URI data type. A URI fragment starts with a hashtag (#) followed by a unique ID value.

Identifiers appear as an "**`id`**" or "**`uuid`**" flag to a data field or assembly. Examples include:

- `<control id="ac-1">`: Uniquely identifies the control.
- `<party uuid="8e83458e-dde5-4ee2-88bc-152f8da3fc31">`: Uniquely identifies the party.

IDs and UUIDs are intended to be managed by tools "behind the scenes," and should not typically be exposed to users for manipulating SSP, SAP, SAR and POA&M content..

An ID reference typically appears with a name and hyphen in front of the "id" (name-id) or "uuid" (name-uuid). It is typically a flag where the relationship is one-to-one, but sometimes a field when the relationship is one-to-many. The name of an ID reference flag/field typically reflects the name of the field to which it points.

Examples include:

- `<responsible-party role-id="prepared-by">`: points to a `role` identified by "prepared-by".
- `<implemented-requirement id="imp-req-01" control-id="ac-2">`: points to the `control` identified by "ac-2".

NIST provides some standard identifiers. Where appropriate, FedRAMP has adopted those and defined additional identifiers as needed. To ensure consistent processing, FedRAMP encourage content creators to use the NIST and FedRAMP-defined identifiers to the greatest degree practical. Deviation is likely to result in processing errors.

2.5.1. Uniqueness of Identifiers

Within FedRAMP deliverables, only `roles` in the `metadata` assembly have ID flags. All other OSCAL identifiers are UUID.

Some role ID values are prescribed by NIST, and others by FedRAMP. These are "reserved" and must not be assigned to other roles for other reasons. The scope of this requirement goes beyond the current OSCAL file to all files in the OSCAL stack as a result of import statements, as described in *Section 2.1, File Content Concepts*.

For UUID fields, if using a tool that properly generates version 4 UUID values, no two will be alike; however, buggy tools have been known to create unexpected duplicate values. In an abundance of caution, tool developers are encouraged to check for unintended duplicates whenever generating a new UUID values. At least during the testing phase of your development lifecycle.

2.5.2. Searching for Information by ID or UUID Values

When searching for an ID or UUID reference, the tool must look both in the current OSCAL file, and other files in the OSCAL stack as described in *Section 2.1, File Content Concepts*.

For example, a UUID reference in an OSCAL-based FedRAMP SAR could refer to a field or assembly in the SAR, SAP, or SSP. XPath and similar standards only search the current file.

This requires OSCAL tools to search the current file first. If the ID or UUID is not found, the tool should follow the file's import statement and search the next file the same way. This must be repeated until either the ID/UUID is found, or all files in the stack have been processed. Of course, tools may limit

Searching for a Field or Assembly by ID or UUID

In general, it is very simple to query for an ID or UUID value within an XML file. Simply use the following XPath query:

```
//*[@id='id-value']
     -OR-
//*[@uuid='uuid-value']
```

Of course, the tool must replace '`id-value`' or '`uuid-value`' above with the appropriate reference.

Ensuring the Field or Assembly Name Matches

Often flags that reference OSCAL information using its ID or UUID will have a name and context that clarifies the expected target. For example, a flag may appear as follows:

```
<field-name component-uuid='1c23ddee-7001-4512-9de1-e062faa69c0a' />
```

To ensure this UUID value points to a component, use the following XPath query:

```
boolean(//*[@uuid='1c23ddee-7001-4512-9de1-e062faa69c0a']/name()='component')
```

If the above expression returns `true`, the UUID points to a component as intended.

Limiting Searches by Field or Assembly Name

Another approach is to limit the search only to fields or flags with a specific name.

The following will only find the UUID value if it is associated with a component. It would work in the SSP, SAP, SAR and POA&M.

```
//component[@uuid='1c23ddee-7001-4512-9de1-e062faa69c0a']
```

The following will only find the UUID value if it is associated with a `component` in the `system-implementation` assembly of the SSP. If the UUID value is

```
/*/system-implementation/component[@uuid='1c23ddee-7001-4512-9de1-
  e062faa69c0a']
```

2.6. Handling of OSCAL Data Types

OSCAL fields and flags have data types assigned to them. NIST provides important information about these data types here:
https://pages.nist.gov/OSCAL/documentation/schema/datatypes/

The following sections describe special handling considerations for data types that directly impact FedRAMP content in OSCAL.

2.6.1. Date and Time in OSCAL Files

Except where noted, all dates and times in the OSCAL-based content must be in an OSCAL dateTime-with-timezone format as documented here:

https://pages.nist.gov/OSCAL/documentation/schema/datatypes/#datetime-with-timezone

This means all dates and times must be represented in the OSCAL file using following format, unless otherwise noted:

`"Y-m-d\TH:i:s.uP"` (See HERE for formatting codes.)

For example, a publication date of 5:30 pm EST, January 10, 2020 must appear as 2020-01-10T17:30:00.00-05:00

This includes:

- Numeric Year: Four-digits
- A dash
- Numeric Month: Two-digit, zero-padded
- A dash
- Numeric Day: Two digit, zero padded
- The capital letter “T” (Do not use lower case)
- Hour: Two digit, zero-padded, 24-hour clock (Use 18 for 6:00 pm)
- A colon
- Minute: Two digit, zero-padded
- A colon
- Seconds: Two digit, zero-padded
- A decimal point
- Fractions of a second: two or three digits, zero padded

Followed by either:

- A capital letter Z to indicate the time is expressed in Coordinated Universal Time (UTC)

OR:

- A plus or minus representing the offset from UTC
- Hour Offset: Difference from UTC, two-digit, padded
- A colon
- Minutes Offset: Difference from UTC, two-digit, padded

This is only for *storing* dates in the OSCAL file. NIST syntax verification tools will generate an error if this format is not found.

Tool developers are encouraged to *present* dates as they have historically appeared in FedRAMP documents. In other words, tools should convert "`2020-03-04T00:00:00.00-05:00`" to "`March 4, 2020`" when presenting the publication date to the user.

Please use the appropriate UTC offset in your region. If you are storing a date and padding the time with zeros, you may also pad the UTC offset with zeros.

2.6.2. UUID Datatypes

NIST typically uses

Any place a UUID flag or UUID reference exits, the datatype is uuid, as required by NIST for OSCAL. NIST requires uuid version 4, as defined by RFC-4122.

Version 4 UUIDs are 128-bit numbers, represented as 32 hexadecimal (base-16) digits in the pattern:

`xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx`
All alphabetic characters (`a-e`) representing hexadecimal values 10 - 15 must be lower case.

Strictly following the RFC ensures the probability of generating an unintended duplicate UUID value is so close to zero it may be ignored. As calculated on Wikipedia, after generating 103 trillion version 4 UUIDs, there is a one in a billion chance of a duplicate.

There are several open-source algorithms and built-in functions for generating UUID. It is important to use one that is known to be fully compliant. Unintended duplicate UUIDs become more likely when using non-compliant or buggy algorithms.

For more information about the use of UUIDs in OSCAL, see *Section 2.5, Assigning Identifiers*.

2.6.3. ID Datatypes

NIST typically uses ID flags in OSCAL where the identifier is intended to be canonical, such as the identifiers for `role` IDs or NIST SP 800-53 controls.

Any place an ID flag or ID reference exits, the datatype is NCName, as defined by the World Wide Web Consortium (W3C). The allowable values of an NCName are limited as follows:

- The first character must be alphabetic [`a-z` or `A-Z`], or an underscore [_].
- The remaining characters must be alphabetic [`a-z` or `A-Z`], numeric [`0-9`], an underscore [_], period [`.`], or dash [-].
- Spaces are not allowed.

NCName values are case sensitive. "`This`" does not match "`this`" in searches.

A tool developer may wish to assign UUID values to ID flags. UUIDs may start with a numeric character, which is invalid for NCName. To ensure a valid datatype when using a UUID value in an ID field, consider prepending "uuid-" to the UUID value.

For more information about the use of IDs in OSCAL, see *Section 2.5, Assigning Identifiers*.

2.6.4. Working With href Flags

All OSCAL-based `href` flags are URIs formatted according to section 4.1 of RFC3986. When assembling or processing an OSCAL-based FedRAMP file, please consider the following:

Absolute Paths: When using an absolute path within a FedRAMP OSCAL file, the path must be publicly accessible from any location on the Internet, to ensure agency and FedRAMP reviewers can reach the information.

Tool developers are encouraged to validate paths before storing or using them in OSCAL files and raise issues to users if paths are not reachable.

Relative Paths: All relative paths are assumed to be based on the location of the OSCAL file, unless tools are explicit as to other handling. Sensitive external documents should travel with the OSCAL file and be linked using a relative path.

Internal Locations: These URI fragments appear as just a hashtag (#) followed by a name, such as `#a3e9f988-2db7-4a14-9859-0a0f5b0eebee`. This notation points to a location internal to the OSCAL content. Typically, this references to a `resource` assembly, but may reference to any field or assembly with a unique ID or UUID.

If only a URI fragment (internal location) is present, the OSCAL tool must strip the hashtag (#) and treat the remaining string as a UUID reference to a resource. The resource may exist in the current OSCAL file, or one of the imported OSCAL files in the stack as described in *Section 2.1, File Content Concepts*.

For example, the following OSCAL content contains a `href` flag with a URI fragment:

URI Fragment Example

```
<system-characteristics>
   <authorization-boundary>
      <diagram uuid="a04b5a0c-6111-420c-9ea3-613629599213">
         <link href="#a3e9f988-2db7-4a14-9859-0a0f5b0eebee"/>
         <caption>Authorization Boundary Diagram</caption>
      </diagram>
   </authorization-boundary>
</system-characteristics>
```

When a tool processes the above example, it should look inside the document for a field or assembly with a UUID of "`a3e9f988-2db7-4a14-9859-0a0f5b0eebee`". This can be accomplished with the following XPath query:

```
//*[@uuid="a3e9f988-2db7-4a14-9859-0a0f5b0eebee"]
```

If this is found to point to a resource assembly, see the *Attachments and Embedded Content in OSCAL Files* section for additional handling.

The name of the field or assembly referenced by the above URI fragment can be determined using the following XPath 2.0 query:

```
//*[@uuid="uri-fragment" | @id="uri-fragment"]/name()
```

To express this in XPath 1.0, you must use the following:

```
name(//*[@uuid="uri-fragment" | @id="uri-fragment"])
```

The above query will return "`resource`", if the URI Fragment references the UUID of a `resource` assembly.

2.6.5. Markup-line and Markup-multiline Fields in OSCAL

As with most machine-readable formats, most of OSCAL's fields are intended to capture short, discrete pieces of information; however, sometimes users require content to be formatted using features such bold, underline, and italics.

For markup-line and markup-multi-line, a subset of HTML is used to format XML-based OSCAL files, while Markdown is used to format JSON-based OSCAL files.

OSCAL provides two types of fields for this purpose:

- **markup-line**: Allows some formatting within a single line of text.
- **Markup-multiline**: Allows all the markup-line formatting, plus allows multiple lines, ordered/ unordered lists, and tables.

In OSCAL-based XML files, markup-line and -multiline uses a subset of HTML.

In OSCAL-based JSON files, markup-line and -multiline uses a subset of markdown.

NIST has implemented only a subset of formatting tags from these standards. This is to ensure formatted content converts completely and consistently between XML and JSON in either direction.

Both *markup-line* and *markup-multiline* support:

- emphasis and important text
- inline code and quoted text
- sub/super-script
- images and links

Markup-multiline also supports:

- Paragraphs
- Headings (Levels 1-6)
- Preformatted text
- Ordered and Unordered Lists
- Tables

For a complete list of markup-line and markup-multiline features, please visit:
https://pages.nist.gov/OSCAL/documentation/schema/datatypes/#markup-data-types

2.6.6. Working with Markup-multiline Content

In JSON, markup-multiline is based on Markdown syntax and requires no special handling. XML-based markup-multiline fields require all content to be enclosed in one of the following tags: `<p>`, `<h1>` - `<h6>`, `<ol>`, `<ul>`, `<pre>`, or `<table>`. At least one of these tags must be present. More than one may be present. All text must be enclosed within one of these tags.

The example below is a common misuse of markup-multiline. The description contains text, but the text is not enclosed in one of the required tags. This will produce an error when checked with the OSCAL schema.

Incorrect Markup-multiline Representation

```
<system-characteristics
   <!-- cut -->
   <description>The xyz system performs ...</description>
</system-characteristics>
```

The simplest way to correct the error is to enclose the text in tags, within the `description` field.

Correct Markup-multiline Representation

```
<system-characteristics
   <!-- cut -->
   <description><p>The xyz system performs ... </p></description>
</system-characteristics>
```

The example below demonstrates a correct use of markup-multiline in XML. Please note, the inclusion of a `<p />` tag on a line by itself inserts an empty paragraph. Within XML and HTML, this is treated as a shortcut, and is interpreted as "`<p></p>`"

Correct Markup-multiline Representation

```
<system-characteristics
   <!-- cut -->
   <description>
      <p>The <b>xyz system</b> performs ... </p>
      <p>The xyz system further supports ... as follows:</p>
      <table>
         <tr>
            <td>Cell A1</td>
            <td>Cell B1</td>
         </tr>
         <tr>
            <td>Cell A2</td>
            <td>Cell B2</td>
         </tr>
      </table>
      <h1>Big Header</h1>
      <p>More detail</p>
      <p><img alt="alt text" src="url" title="title text"/></p>
   </description>
</system-characteristics>
```

For more information, please visit:
https://pages.nist.gov/OSCAL/documentation/schema/datatypes/#markup-data-types

2.6.7. Special Characters in OSCAL

While OSCAL itself does not directly impose special character handling requirements, XML and JSON do. Characters, such as ampersand (&), greater than (>), less than (<), and quotes (") require special treatment in OSCAL files, depending on the format. For a complete list of special characters and the appropriate treatment for each format, please visit:

https://pages.nist.gov/OSCAL/documentation/schema/datatypes/#specialized-character-mapping

2.7. Citations and Attachments in OSCAL Files

OSCAL is designed so that all citations and attachments are defined once at the end of the file as a `resource`, and then referenced as needed throughout the file. This includes logos, diagrams, policies, procedures, plans, evidence, and interconnection security agreements (ICAs).

Each `resource` may be referenced from anywhere in the OSCAL file, using its resource UUID.

```
<back-matter>
   <resource uuid="3df7eeea-421b-459d-98cf-3d972bec610a">
      <title>Attachment or Document Title</title>
      <desc>An optional description of the attachment.</desc>
      <rlink href="./relative/path/doc.pdf" media-type="application/pdf" />
      <rlink href="//absolute/path/doc.pdf" media-type="application/pdf" />
      <base64            filename="doc.pdf" media-type="application/pdf">
         00000000
      </base64>
   </resource>
</back-matter>
```

The `media-type` flag should be present, and must be set to an Internet Assigned Numbers Authority (IANA) Media Type.

2.7.1. Citations

Citations are for reference. The content is not included with the authorization package.

Citations use an `rlink` field with an *absolute path* to content that is accessible by FedRAMP and Agency reviewers from government systems.

Examples include links to publicly available laws such as FISMA, and applicable standards, such NIST SP 800 series documents.

2.7.2. Attachments

Unlike citations, attachments are included with the authorization package when submitted.

Tools may either embed an attachment using the `base64` field, or point to an attachment using an `rlink` field. If no base64 embedded content is present, at least one rlink field must exist with either an *absolute path* to content that is accessible by FedRAMP and Agency reviewers from government systems, or a relative path.

If a relative path is used, the attachment must be delivered with the OSCAL file and packaged such that the attachment exists in the correct relative location.

Examples include the logo for the CSP or 3PAO, authorization boundary diagrams, policies, process, plans, raw scanner output, assessment evidence, and POA&M deviation request evidence.

Tools should embed (base64) or link to (rlink) an attachment once as a `resource` in `back-matter`, then use URI fragments to reference the attachment anyplace it is needed within the body of the OSCAL file, as described in *Section 2.5, Assigning Identifiers* or *Section 2.6.2, Working With href Flags*.

For example, a policy document that satisfies several control families is attached as a `resource` in the `back-matter`, with a UUID of "`3df7eeea-421b-459d-98cf-3d972bec610a`". Each control satisfied by that policy, links to the policy using a URI fragment as follows:

```
<link href="#3df7eeea-421b-459d-98cf-3d972bec610a" rel="policy" />
```

As described in *Section 2.6.2, Working With href Flags*, when a tool identifies a URI fragment in an `href` value, the leading hashtag (#) must be dropped and the remaining value is expected to reference an OSCAL addressable ID or UUID. This ID/UUID may be either within the OSCAL file itself, or within other OSCAL documents linked via the `import` field.

The policy's title, version, publication date and other details are defined once in the `resource`, and are displayed anyplace the policy is referenced. If a newer policy is published, only the one `resource` in needs to be updated.

If the policy's location is identified as "`./policies/doc.pdf`", the OSCAL file and the doc.pdf file should be delivered together and packaged such that the folder containing the oscal file includes a sub-folder named "`policies`", which contains the "`doc.pdf`" file.

When using attachments with relative paths, consider using a technology such as a ZIP archive to package and deliver attachments while maintaining their relative position to the OSCAL file.

2.7.3. Including Multiple rlink and base64 Fields

Within a `resource`, there may be:

- no `rlink` nor `base64` fields;
- one or more `rlink` fields;
- one or more `base64`-encoded data fields within the OSCAL file; or
- any combination of `rlink` and `base64` fields.

OSCAL allows multiple `rlink` and `base64` fields to exist within the same `resource`. This provides the flexibility to identify multiple locations or multiple formats of the same resource. Some examples of using multiple rlink and/or base64 fields within the same resource include:

- **Multiple Locations**: Multiple `rlink` fields allow an OSCAL tool to include one `rlink` field with an *absolute* path to the authoritative location of a policy document within the CSP's intranet. The same `resource` could have a second `rlink` field with a *relative* path to the same policy document. Having both allows the CSP to maintain the link to authoritative location of the policy when working with the OSCAL file internally, while allowing a cached, local copy to travel with the OSCAL file when delivered to FedRAMP for review.
- **Multiple Quality Levels**: Multiple `rlink` or `base64` fields allow both low-resolution and high-resolution versions of the same image, which is sometimes used to boost the performance of web-based applications.
- **Multiple Formats**: Multiple `rlink` or `base64` fields allow a portable network graphic (PNG) version of an image may be provided for presentation by a web application, and a more detailed portable document format (PDF) version of the same image for download by users.

2.7.4. Handling Multiple rlink and base64 Fields

NIST designed `resource` assemblies to be flexible, and wanted to offer developers the flexibility to implement handling of multiple `rlink` and `base64` fields on a case-by-case basis.

This section describes FedRAMP's processing of multiple `rlink` and `base64` fields, which will be used by default unless a compelling circumstance requires otherwise.

FedRAMP prefers authorization packages use `base64` embedded content for diagrams and graphics, but accepts both `base64` and `rlink` options. FedRAMP prefers documents, such as policies and plans, are attached using `rlink` fields and relative paths.

If the tool is designed to work interactively with a user, the tool developer should consider designing the tool to make intelligent choices based on context and circumstances where practical. The tool could also present valid choices to the user where the correct choice is ambiguous to the tool.

When more than one rlink and/or base64 field is present in a resource, FedRAMP's automated processing tools will attempt to find valid content using the following priority, unless specified otherwise:

1. FedRAMP tools will look first in `base64` fields
    a. Start with the first `base64` field in the resource.
    b. Check each `base64` field in the sequence in which they appear in the `resource`.
    c. If valid content is found, stop looking and use the content.
    d. If no valid content is found after checking all `base64` fields in the resource, proceed to step #2.
2. If no valid base64 content is found, look in `rlink` fields.
    a. Start with the first `rlink` field in the resource.
    b. Check each `rlink` field in the sequence in which they appear in the `resource`.
    c. If valid content is found, stop looking and use the content.
    d. If no valid content is found after checking all `rlink` fields, treat the resource as invalid, which may include raising a warning or error message.

2.7.5. Citation and Attachment FedRAMP Extensions

While NIST designed the resource assembly to be robust, some information required by FedRAMP must be provided using a FedRAMP extension.

The FedRAMP extension, "type" may be used within any resource, and must be used to identify policies, plans, evidence and other types as described in relevant sections of these guides.

For policies, plans, user guides, and other documents, FedRAMP requires the document's title, version, and publication date. While `resource` includes a title field, it does not include version or publication date fields. The FedRAMP extensions, "version" and "publication" must be used to capture this information.

The following demonstrates the use of FedRAMP extensions within a resource.

```
<back-matter>
   <resource uuid="3df7eeea-421b-459d-98cf-3d972bec610a">
      <title>Attachment or Document Title</title>
        <prop name="type" ns="https://fedramp.gov/ns/oscal">policy</prop>
        <prop name="version" ns="https://fedramp.gov/ns/oscal">2.1</prop>
        <prop name="publication"
              ns="https://fedramp.gov/ns/oscal">2018-11-11T00:00:00Z</prop>
      <rlink href="./relative/path/doc.pdf" media-type="application/pdf" />
      <rlink href="//absolute/path/doc.pdf" media-type="application/pdf" />
      </base64>
   </resource>
</back-matter>
```

2.7.6. Citation and Attachment Conformity Tags

In most cases, `resource` assemblies containing attachments or citations are present to satisfy FedRAMP requirements. As described in *Section 3.1.2, FedRAMP Conformity Tagging*, where required FedRAMP content is clearly identified by OSCAL syntax alone, no special handling is required; however, in some cases, FedRAMP requires a conformity tag to ensure FedRAMP tools are able to clearly identify required content.

The following represents the use of FedRAMP conformity tag within a resource.

```
<back-matter>
     <resource uuid="e552fb72-d662-4c01-b2d7-4dcb2086bb07">
        <title>Penetration Test Report</title>
        <prop name="conformity"
         ns="https://fedramp.gov/ns/oscal">penetration-test-report</prop>
        <prop name="type" ns="https://fedramp.gov/ns/oscal">report</prop>
        <rlink media-type="application/pdf"
                    href="./pen_test_report.pdf" />
   </resource>
</back-matter>
```

3. FEDRAMP EXTENSIONS, CONFORMITY TAGS, DEFINED IDENTIFIERS, AND ACCEPTED VALUES

NIST designed the core OSCAL syntax to meet model cybersecurity information that is common to any organization and compliance framework. They recognized that each framework and organization may have unique needs. Instead of trying to provide a language that meets each of those unique needs, NIST gave organizations the ability to tailor OSCAL to address specific needs.

A summary of the FedRAMP extensions, conformity tags, defined identifiers, and accepted values appears in the FedRAMP OSCAL Registry.

FedRAMP has tailored OSCAL by specifying:

- **Extensions**: allow FedRAMP's OSCAL-based content to capture information that is not available in the core OSCAL syntax.
- **Conformity Tags**: This is a special extension that allows FedRAMP processing tools to find required information without human intervention.
- **Defined Identifiers**: FedRAMP defines several unique ID values that must be used in the ID flag of certain OSCAL fields and assemblies.
- **Accepted Values**: For many fields, FedRAMP specifies a case-sensitive set of values. Only these values are recognized by FedRAMP processing tools.

3.1.1. FedRAMP Extensions

All FedRAMP extensions include a namespace (`ns`) flag set to `"https://fedramp.gov/ns/oscal"`.

There are several pieces of information required in FedRAMP templates that cannot be modeled using OSCAL's core syntax. NIST wanted to limit the core OSCAL syntax to those elements that are universal across most cybersecurity frameworks. They designed OSCAL to be extended where unique needs existed.

NIST allows organizations to extend OSCAL anyplace `prop` fields, `part` assemblies, or `annotation` assemblies exist in the core syntax. (Please note, there are currently no `part` assemblies in the SSP, SAP, SAR, or POA&M.) There are two fundamental requirements for extending OSCAL:

- The organization must establish a unique namespace (ns) identifier, such as (`ns="http://domain.tld/ns/oscal"`), and use it to consistently tag all `prop`, `part`, and `annotation` extensions from that organization.
- The organization is responsible for defining, managing, and communicating all names (`name="scan-type"`) defined and tagged with the above name space identifier.

NIST's core OSCAL `prop` fields and `annotation` assemblies have no `ns` flag. If an `ns` flag is present, it is an organization-defined extension. This allows each industry standards body or organization to create their own extensions in their own name space without concern for overlapping names.

The above approach ensures two different organizations can create their own extensions without concern for reusing the same name values. At some point in the future, NIST may provide a registry for organizational extensions. For now, FedRAMP is publishing its own registry to document its extensions.

All FedRAMP extensions include a namespace (`ns`) flag set to "`https://fedramp.gov/ns/oscal`".

For example, if the core OSCAL syntax has a status field, but both FedRAMP and the payment card industry (PCI) require their own framework-specific `status` field, each may define an extension with the `name="status"`, and assign their own `ns` flag. This results in three possible status fields as follows:

NIST OSCAL User Representation
`<prop name="status">There is no @ns, so this is core OSCAL syntax</prop>`
XPath Query
`//prop[@name="user"][not(@ns)]`

When searching an OSCAL file for a `prop` or `annotation` extensions that is part of the core OSCAL syntax, developers must filter out any with an `ns` flag using the syntax above.

FedRAMP User Representation
`<prop name="status" ns="https://fedramp.gov/ns/oscal">FedRAMP Status</prop>`
XPath Query
`//prop[@name="status"][@ns="https://fedramp.gov/ns/oscal"]`

(Possible) PCI User Representation
`<prop name="user" ns="https://pcisecuritystandards.org/ns/oscal">PCI User</prop>`
XPath Query
`//prop[@name="user"][@ns="https://pcisecuritystandards.org/ns/oscal"]`

* This is an example, and is not intended to represent an actual PCI extension.

Tool developers must always refer to extensions using **both** the `name` and `ns` flags as a pair.

All FedRAMP extensions will appear as either:

```
<prop name="____" ns="https://fedramp.gov/ns/oscal">Value</prop>
```

or:

```
<annotation name="____" ns="https://fedramp.gov/ns/oscal" value="Value">
   <remarks><p>A possible remark about the value</p></remarks>
</annotation>
```

NOTE: The catalog and profile OSCAL models also allow the `part` assembly to be used for extensions. This is not currently the case for the OSCAL SSP, SAP, SAR, or POA&M.

FedRAMP extensions are cited in relevant portions of this document and summarized in the FedRAMP OSCAL Registry.

3.1.2. FedRAMP Conformity Tagging

Some FedRAMP-required content is easily found by processing tools using the core OSCAL syntax; however, OSCAL's versatility makes other required FedRAMP content difficult to identify or ambiguous.

FedRAMP Conformity Tagging is a special extension that allows FedRAMP processing tools to find required information without human intervention, where the core OSCAL syntax alone is not sufficient for validation.

For example, a contingency plan is typically provided in OSCAL as a `resource` attachment in the `back-matter` portion of an OSCAL file. The attachment would include a conformity tag with a value of "`contingency-plan`". This ensures FedRAMP tools know which attachment is the CSP's contingency plan.

Unlike defined identifiers (below), FedRAMP conformity tags are designed to co-exist with conformity tags from other organizations and compliance frameworks.

FedRAMP Conformity Tags are cited in relevant portions of this document and summarized in the FedRAMP OSCAL Registry.

3.1.3. OSCAL and FedRAMP-Defined Identifiers

In limited portions of the OSCAL syntax, such as roles, NIST defines *case-sensitive* identifiers common to most OSCAL use cases.

FedRAMP carefully expands this list for cloud-base systems, and NIST is evaluating the FedRAMP additions for inclusion in core OSCAL. As of this publication, NIST views all identifiers as draft. As such, they are subject to change until OSCAL 1.0 is released later in calendar year 2020.

For example, every system must have a system owner, and typically has an Information System Security Officer (ISSO). NIST defines specific identifiers for these roles, which enables tools to easily identify the individuals associated with these roles.

FedRAMP defined identifiers are cited in relevant portions of this document and summarized in the FedRAMP OSCAL Registry.

3.1.4. OSCAL and FedRAMP Accepted Values

To facilitate consistent processing, the value for some fields is limited to a list of *case-sensitive* acceptable values. For some fields, OSCAL defines acceptable values, which are enforced by OSCAL-based syntax validation mechanisms.

There are additional fields, where OSCAL syntax validation mechanisms will accept any value, but FedRAMP provides a limited list of *case-sensitive* acceptable values. Where defined, only these values are recognized by FedRAMP processing tools.

For example, every control requires an implementation status. FedRAMP only accepts one of five possible responses for this status, which must be provided using one of the specified choices.

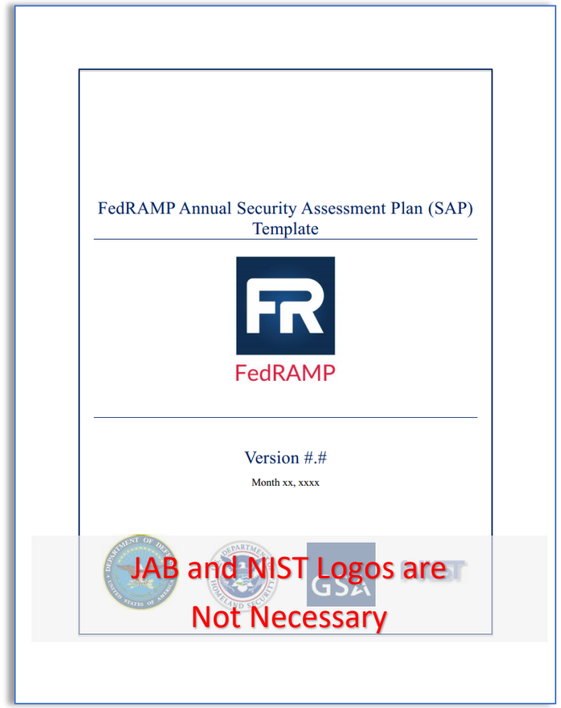
FedRAMP accepted values are cited in relevant portions of this document and summarized in the FedRAMP OSCAL Registry.

4. EXPRESSING COMMON FEDRAMP TEMPLATE ELEMENTS IN OSCAL

While each FedRAMP template has a unique purpose, they share common information elements, such as title and publication date. These common elements are expressed using the same OSCAL syntax for the SSP, SAP, SAR, and POA&M. This section provides OSCAL syntax for these common elements, including:

- Document Basics:
  - Document Title
  - Document Publication Date
  - Document Version
  - Document Sensitivity & Ownership Markings
  - Prepared By
  - Prepared For
  - Document Revision History
- OSCAL-Additional Document Basics:
  - Last Modified Date
  - OSCAL Syntax Version
- Logos:
  - FedRAMP Logo
  - CSP Logo
  - Assessor Logo
  - Consulting 3PAO Logo
- Attachments:
  - FedRAMP Acronyms
  - FedRAMP Citations (Laws, Regulations, Standards, and Guidance)

The following pages are formatted for tabloid (11" x 17") paper in landscape orientation.



The **FedRAMP Logo** is base 64 encoded in the back-matter section of the OSCAL-based FedRAMP Templates, and can be referenced with the following XPath:

```
/*/back-matter/resource/prop[@name='conformity'][@ns='https://fedramp.gov/ns/oscal']
  [string(.)='fedramp-logo']/../base64
```

4.1. Title Page

Representation

```
<metadata>
   <title>FedRAMP System Security Plan (SSP)</title>
   <published>2020-06-01T00:00:00.00-04:00</published>
   <last-modified>2020-05-31T00:00:00.00-04:00</last-modified>
   <version>0.0</version>
   <oscal-version>1.0-Milestone3</oscal-version>
</metadata>

<!-- OSCAL File Body -->

<back-matter>
   <resource uuid="uuid-v4-cut-0000000">
      <desc>FedRAMP Logo</desc>
      <prop name='conformity' ns='https://fedramp.gov/ns/oscal'>fedramp-logo</prop>
      <rlink href="" />
      <base64 filename="FedRAMP_LOGO.png">
      <!-- Base64-encoded Logo Cut -->00000000
      </base64>
   </resource>
</back-matter>
```

FedRAMP-Conformity Tag
FedRAMP Logo:
- `fedramp-logo`

XPath Queries

```
Document Title:
  /*/metadata/title

Document Published Version #:
  /*/metadata/version

Document Published Date (will need to convert data for presentation):
  /*/metadata/published

FedRAMP's Logo:
  /*/back-matter/resource/prop[@name='conformity'][@ns='https://fedramp.gov/ns/oscal']
  [string(.)='fedramp-logo']/../base64

Document Sensitivity Label (may be more than one):
  /*/metadata/prop[@name="marking"]
```

NOTES:

- There may be more than one *Document Sensitivity Label* if needed.
  - Tools should display and/or notify the user of all sensitivity markings.
- The logos on older FedRAMP Templates are not necessary. This includes the NIST Logo, as well as the three Joint Authorization Board (JAB) Agency Logos.

FEDRAMP SYSTEM SECURITY PLAN (SSP) _____ BASELINE TEMPLATE

CSP Name | Information System Name *Version #.#, Date*

SYSTEM SECURITY PLAN

Prepared by

Identification of Organization that Prepared this Document		
	Organization Name	<Enter Company/Organization>.
	Street Address	<Enter Street Address>
	Suite/Room/Building	<Enter Suite/Room/Building>
	City, State Zip	<Enter Zip Code>

Prepared for

Identification of Cloud Service Provider		
	Organization Name	<Enter Company/Organization>.
	Street Address	<Enter Street Address>
	Suite/Room/Building	<Enter Suite/Room/Building>
	City, State Zip	<Enter Zip Code>

See Next Page for "Prepared for"

<CSP> FedRAMP Annual SAP Template *<Date of modification>*

Prepared by

Identification of IA that Prepared this Document		
<insert logo>	Organization Name	
	Street Address	
	Suite/Room/Building	
	City, State Zip	

Prepared for

Identification of Cloud Service Provider		
<insert logo>	Organization Name	
	Street Address	
	Suite/Room/Building	
	City, State Zip	

See Next Page for "Prepared for"

4.2. Prepared By (Third Party)

The FedRAMP SAP and SAR must always indicate the assessing organization using this Prepared By syntax.

Representation

```
<metadata>
   <role id="prepared-by">
      <title>Prepared By</title>
      <desc>The organization that prepared this content.</desc>
   </role>
   <!-- cut: other role assemblies-->
   <!-- cut: location assemblies-->
   <party uuid="f84d8edc-d83e-440d-96c9-09b28c395ad5">
      <org>
         <org-name>Name of Consulting Org</org-name>
         <short-name>Acronym/Short Name</short-name>
         <address>
            <!-- address lines cut here for space -->
         </address>
      </org>
   </party>
   <!-- cut: other party assemblies -->
   <responsible-party role-id="prepared-by">
      <party-id>f84d8edc-d83e-440d-96c9-09b28c395ad5</party-id>
   </responsible-party>
</metadata>

<!-- OSCAL File Body -->

<back-matter>
   <resource id="1507f5e0-635c-4e23-a5f3-93f368f8e022">
      <desc>Preparer Logo</desc>
      <prop name='conformity' ns='https://fedramp.gov/ns/oscal'>prepared-by-logo</prop>
      <!-- Use rlink and/or base64 -->
      <rlink href="./party-1-logo.png" media-type="image/png" />
      <base64>00000000</base64>
   </resource>
</back-matter>
```

NIST-Defined Identifier

Required Role ID:

- `prepared-by`

FedRAMP-Conformity Tag

Preparer's Logo:

- `prepared-by-logo`

XPath Queries

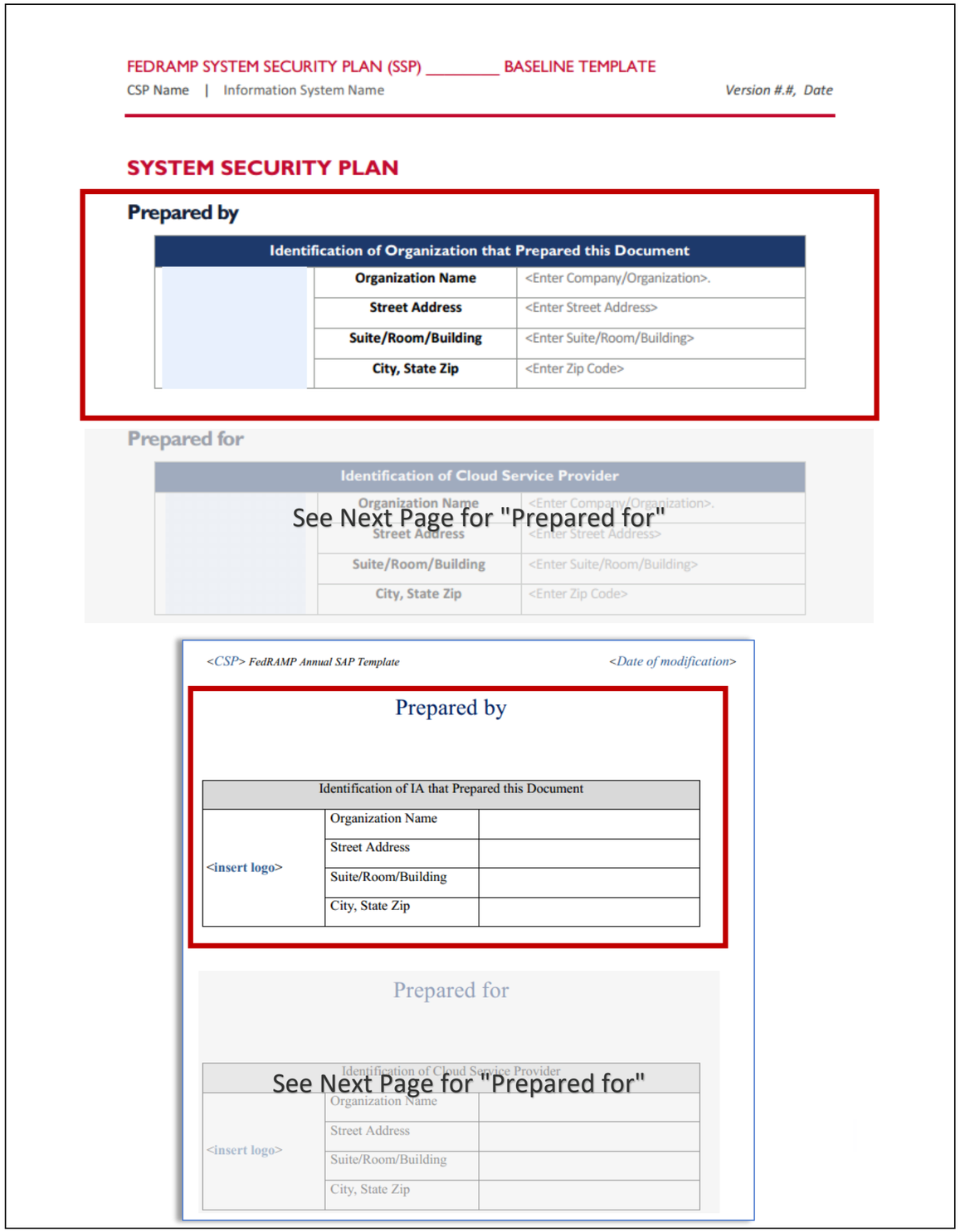
```
Preparer Organization's Logo:
  /*/back-matter/resource/prop[@name='conformity'][@ns='https://fedramp.gov/ns/oscal']
  [string(.)='prepared-by-logo']/../base64          OR
  /*/back-matter/resource/prop[@name='conformity'][@ns='https://fedramp.gov/ns/oscal']
  [string(.)='prepared-by-logo']/../rlink/@href

Preparer Organization's Name and Address:
  /*/metadata/party[@id=[/*/metadata/responsible-party[@role-id='prepared-by']/party-
  id]]/org/org-name

NOTE: Replace "org-name" with "address/addr-line", "address/city", "address/state", or
  "address/zip" as needed. There may be more than one addr-line.
```

NOTES:

- The `responsible-party` assembly connects the role to the party and is required for compliance.
- If the content was prepared by an organization other than the CSP, their logo should also appear in back-matter.



FEDRAMP SYSTEM SECURITY PLAN (SSP) _____ BASELINE TEMPLATE

CSP Name | Information System Name *Version #.#, Date*

SYSTEM SECURITY PLAN

Prepared by

Identification of Organization that Prepared this Document		
	Organization Name	<Enter Company/Organization>.
	Street Address	<Enter Street Address>
	Suite/Room/Building	<Enter Suite/Room/Building>
	City, State Zip	<Enter Zip Code>

Prepared for

Identification of Cloud Service Provider		
	Organization Name	<Enter Company/Organization>.
	Street Address	<Enter Street Address>
	Suite/Room/Building	<Enter Suite/Room/Building>
	City, State Zip	<Enter Zip Code>

See Next Page for "Prepared for"

<CSP> FedRAMP Annual SAP Template *<Date of modification>*

Prepared by

Identification of IA that Prepared this Document		
<insert logo>	Organization Name	
	Street Address	
	Suite/Room/Building	
	City, State Zip	

Prepared for

Identification of Cloud Service Provider		
<insert logo>	Organization Name	
	Street Address	
	Suite/Room/Building	
	City, State Zip	

See Next Page for "Prepared for"

4.3. Prepared By (CSP Self-Prepared)

This is applicable where the CSP creates or updates its own SSP or POA&M content. The FedRAMP SAP and SAR must never be CSP self-prepared.

Representation

```
<metadata>
   <role id="prepared-by">
      <title>Prepared By</title>
      <desc>The organization that prepared this content.</desc>
   </role>
   <!-- cut: other role assemblies-->
   <!-- cut: location assemblies-->
   <party id="2e0db7cf-08f5-472e-9360-fb3a9698476d">
      <org>
         <org-name>Cloud Service Provider (CSP) Name</org-name>
         <short-name>CSP Acronym/Short Name</short-name>
         <location-id>location-1</location-id>
      </org>
   </party>
   <!-- cut: other party assemblies -->
   <responsible-party role-id="prepared-by">
      <party-id>c2e0db7cf-08f5-472e-9360-fb3a9698476dsp</party-id>
   </responsible-party>
</metadata>

<!-- OSCAL File Body -->

<back-matter>
   <resource id="74a61c36-ad18-4ee3-8bc4-6e2f917b0ce8">
      <desc>CSP Logo</desc>
      <prop name='conformity' ns='https://fedramp.gov/ns/oscal'>prepared-by-logo</prop>
      <prop name="conformity" ns="https://fedramp.gov/ns/oscal">csp-logo</prop>
      <!-- Use rlink and/or base64 -->
      <rlink href="./logo.png" media-type="image/png" />
      <base64>00000000</base64>
   </resource>
</back-matter>
```

NIST-Defined Identifier

Required Role ID:

- `prepared-by`

FedRAMP-Conformity Tag

Preparer's Logo:

- `prepared-by-logo`

XPath Queries

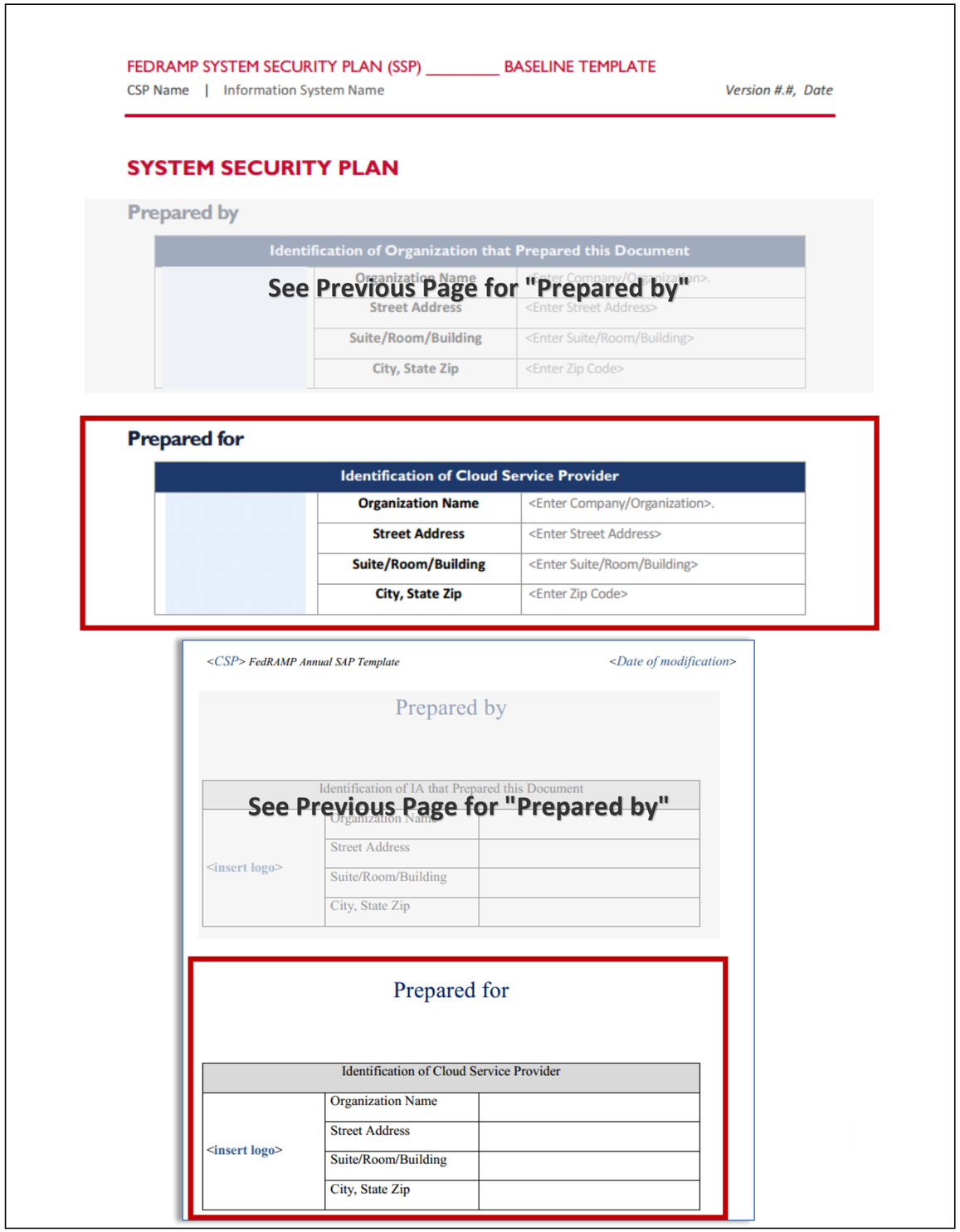
```
Preparer Organization's Logo:
  /*/back-matter/resource/prop[@name='conformity'][@ns='https://fedramp.gov/ns/oscal']
  [string(.)='prepared-by-logo']/../base64          OR
  /*/back-matter/resource/prop[@name='conformity'][@ns='https://fedramp.gov/ns/oscal']
  [string(.)='prepared-by-logo']/../rlink/@href

Preparer Organization's Name and Address:
  /*/metadata/party[@id=[/*/metadata/responsible-party[@role-id='prepared-by']/party-
  id]]/org/org-name

NOTE: Replace "org-name" with "address/addr-line", "address/city", "address/state", or
  "address/zip" as needed. There may be more than one addr-line.
```

NOTES:

- The `responsible-party` assembly connects the role to the party and is required for compliance.
- If the content was prepared by the CSP, the CSP's logo should contain two conformity tags: one to designate the "csp-logo", and another to designate the "prepared-by-logo".



4.4. Prepared For (CSP)

For FedRAMP SSP, SAP, SAR, and POA&M, the "Prepared For" is typically the CSP; however, it may be different if an unforeseen circumstance requires another party to be named. For this reason, "Prepared For" and CSP have separately defined roles.

Representation

```
<metadata>
   <!-- prop -->
   <role id="prepared-for">
      <title>Prepared For</title>
      <desc>The CSP for FedRAMP SSP, SAP, SAR, and POA&M.</desc>
   </role>
   <role id="cloud-service-provider">
      <title>Cloud Service Provider</title>
      <short-name>CSP</short-name>
   </role>
   <!-- cut: other role assemblies-->
   <!-- cut: location assemblies-->
   <party id="2e0db7cf-08f5-472e-9360-fb3a9698476d">
      <org>
         <org-name>Cloud Service Provider (CSP) Name</org-name>
         <short-name>CSP Acronym/Short Name</short-name>
         <location-id>location-1</location-id>
      </org>
   </party>
   <!-- cut: other party assemblies -->
   <responsible-party role-id="prepared-for">
      <party-id>2e0db7cf-08f5-472e-9360-fb3a9698476d</party-id>
   </responsible-party>
</metadata>
<!-- OSCAL File Body -->
<back-matter>
   <resource id="74a61c36-ad18-4ee3-8bc4-6e2f917b0ce8">
      <desc>CSP Logo</desc>
      <prop name="conformity" ns="https://fedramp.gov/ns/oscal">prepared-for-logo</prop>
      <prop name="conformity" ns="https://fedramp.gov/ns/oscal">csp-logo</prop>
      <!-- Use rlink and/or base64 -->
      <rlink href="./logo.png" media-type="image/png" />
      <base64>00000000</base64>
   </resource>
</back-matter>
```

NIST-Defined Identifiers

Required Role IDs:

- prepared-for
- cloud-service-provider

FedRAMP-Conformity Tags

Prepared For Logo:

- prepared-for-logo

CSP's Logo:

- csp-logo

XPath Queries

CSP's Logo:

```
/*/back-matter/resource/prop[@name='conformity'][@ns='https://fedramp.gov/ns/oscal']
[string(.)='csp-logo']/../base64          OR
/*/back-matter/resource/prop[@name='conformity'][@ns='https://fedramp.gov/ns/oscal']
[string(.)='csp-logo']/../rlink/@href
```

Prepared For (CSP) Details:

```
/*/metadata/party[@id=[/*/metadata/responsible-party[@role-id='prepared-for']/party-
id]]/org/org-name
```

Prepared For Details:

```
/*/metadata/party[@id=[/*/metadata/responsible-party[@role-id='prepared-for']/party-
id]]/org/org-name
```

NOTE: Replace "org-name" with "address/addr-line", "address/city", "address/state", or "address/zip" as needed. There may be more than one addr-line.

FEDRAMP SYSTEM SECURITY PLAN (SSP) _____ BASELINE TEMPLATE

CSP Name | Information System Name *Version #.#, Date*

DOCUMENT REVISION HISTORY

Date	Description	Version	Author
<Date>	<Revision Description>	<Version>	<Author>
<Date>	<Revision Description>	<Version>	<Author>
<Date>	<Revision Description>	<Version>	<Author>

The `remarks` field is *Markup multiline*, which enables the text to be formatted. This requires special handling. See *Section 2.5.3 Markup-line and Markup-multiline Fields in OSCAL*, or visit: https://pages.nist.gov/OSCAL/documentation/schema/datatypes/#markup-multiline

NOTE: At time of publication, NIST is evaluating the possibility of including party-uuid or similar in the revision assembly. This section will be updated if that decision is made.

4.5. Document Revision History

The OSCAL revision history requires one FedRAMP extension to fully meet FedRAMP's revision history requirements.

Representation

```
<metadata>
   <!-- title, published, last-modified, version, oscal-version -->
   <revision-history>
      <revision>
         <published>2019-06-01T00:00:00.00-04:00</published>
         <version>1.0</version>
         <oscal-version>1.0-Milestone3</oscal-version>
         <prop name="party-uuid"
      ns="https://fedramp.gov/ns/oscal">f84d8edc-d83e-440d-96c9-09b28c395ad5</prop>
         <remarks><p>Initial publication.</p></remarks>
      </revision>
      <revision>
         <published>2020-06-01T00:00:00.00-04:00</published>
         <version>2.0</version>
         <oscal-version>1.0-Milestone3</oscal-version>
         <prop name="party-uuid"
      ns="https://fedramp.gov/ns/oscal">2e0db7cf-08f5-472e-9360-fb3a9698476d</prop>
         <remarks><p>Updated for annual assessment.</p></remarks>
      </revision>
      <!-- Additional revision assemblies as needed. -->
   </revision-history>
   <!-- doc-id, prop, link, role -->
<metadata>
```

FedRAMP Extension (Author)

prop (`ns="https://fedramp.gov/ns/oscal"`):

- `name="party-id"`

XPath Queries

```
Number of Revision Entries:
  count(/*/metadata/revision-history/revision)

Revision Date for Individual Entry:
  /*/metadata/revision-history/revision[1]/published

Description for Individual Entry:
  /*/metadata/revision-history/revision[1]/remarks/string()

Version for Individual Entry:
  /*/metadata/revision-history/revision[1]/version

Author for Individual Entry:
  /*/metadata/party[@uuid=/*/metadata/revision-history/revision[1]/prop [@name='party-
  uuid'][@ns='https://fedramp.gov/ns/oscal']]/org/short-name
```

Replace "[1]" with "[2]", "[3]", etc.

NOTES:

- The Revision History's Author field is addressed using a FedRAMP extension, which points to a metadata `party`.
- The published field requires the OSCAL data type, dateTime-with-timezone.
- FedRAMP only requires the publication date, not the time.
  - The time portion may be replaced with all zeros.
  - FedRAMP tools should present only the date, and use a more user-friendly format.

FEDRAMP SYSTEM SECURITY PLAN (SSP) _____ BASELINE TEMPLATE

CSP Name | Information System Name *Version #.#, Date*

DOCUMENT REVISION HISTORY

Date	Description	Version	Author
<Date>	<Revision Description>	<Version>	<Author>
<Date>	<Revision Description>	<Version>	<Author>
<Date>	<Revision Description>	<Version>	<Author>

How to contact us

For questions about FedRAMP, or for technical questions about this document including how to use it, contact *info@FedRAMP.gov*

For more information about the FedRAMP project, see www.FedRAMP.gov

4.6. How to Contact Us

The FedRAMP email and web site addresses are part of the organizational content for the FedRAMP PMO party. This information already exists in OSCAL-based FedRAMP Templates.

There must be a `role` defined in the file with the ID value set to "`fedramp-pmo`". There must be a `party` defined with FedRAMP's details, and there must be a `responsible-party` defined, linking the "`fedramp-pmo`" `role-id` to the FedRAMP `party` uuid.

Representation

```
...<metadata>

      <role id="fedramp-pmo">
         <title>FedRAMP Program Management Office</title>
      </role>

      <!-- location -->

      <party uuid="77e0e2c8-2560-4fe9-ac78-c3ff4ffc9f6d" type="organization">
         <party-name>FedRAMP Program Management Office</party-name>
         <short-name>FedRAMP PMO</short-name>
         <link href="https://fedramp.gov" />
         <address type="work">
            <addr-line>1800 F St. NW</addr-line>
            <addr-line/>
            <city>Washington</city>
            <state>DC</state>
            <postal-code/>
            <country>US</country>
         </address>
         <email>info@fedramp.gov</email>
      </party>

      <responsible-party role-id="fedramp-pmo">
         <party-uuid>77e0e2c8-2560-4fe9-ac78-c3ff4ffc9f6d</party-uuid>
      </responsible-party>
...<metadata>
```

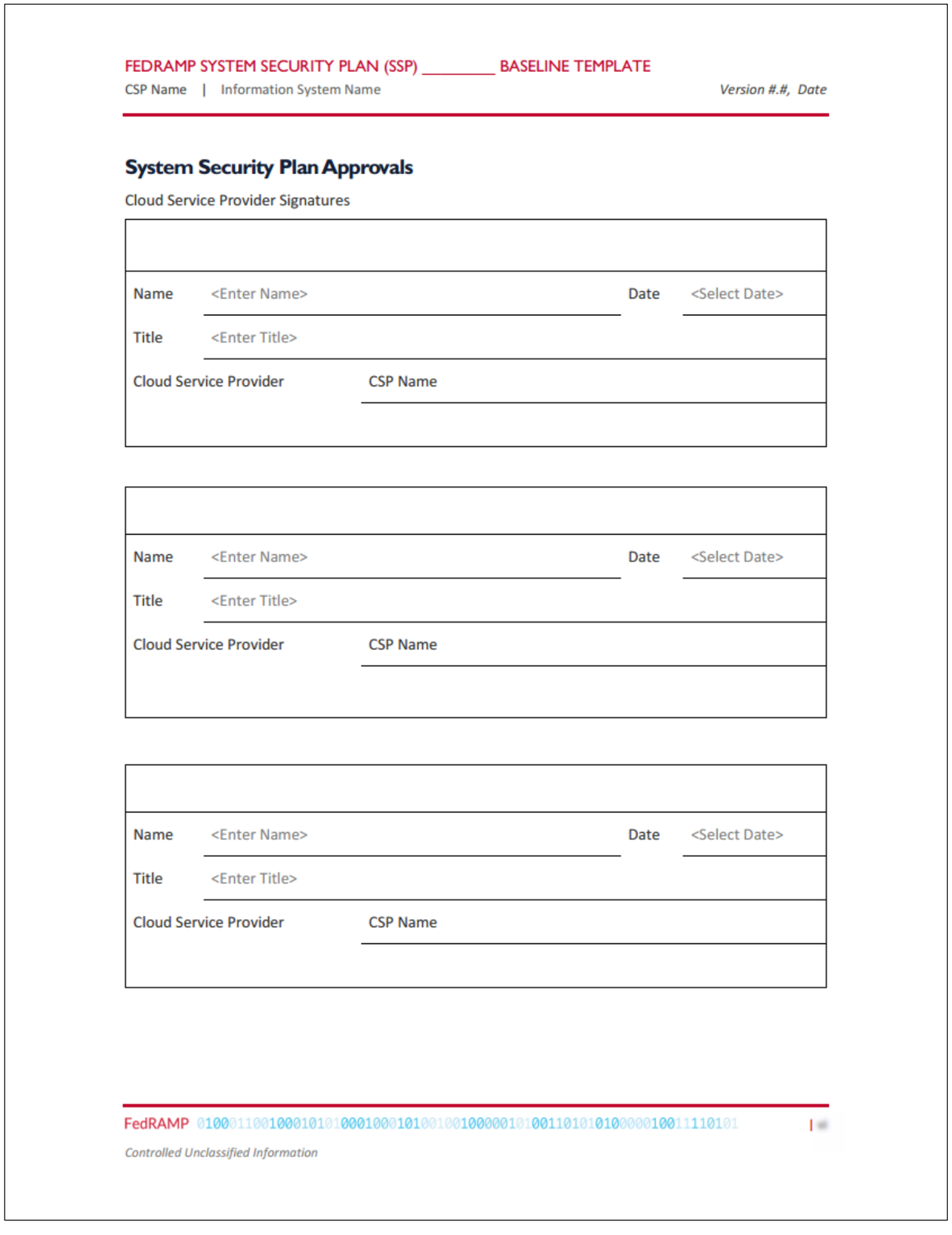
FedRAMP-Role Identifiers:

- `fedramp-pmo`

XPath Queries

```
FedRAMP Email Address:
  /*/metadata/party[@uuid=/*/metadata/responsible-party[@role-id='fedramp-pmo']
  /party-uuid]/email

FedRAMP Web Site Address:
  /*/metadata/party[@uuid=/*/metadata/responsible-party[@role-id='fedramp-pmo']
  /party-uuid]/link/@href
```



FEDRAMP SYSTEM SECURITY PLAN (SSP) _____ BASELINE TEMPLATE

CSP Name | Information System Name *Version #.#, Date*

System Security Plan Approvals

Cloud Service Provider Signatures

Name	<Enter Name>	Date	<Select Date>
Title	<Enter Title>		
Cloud Service Provider	CSP Name		

Name	<Enter Name>	Date	<Select Date>
Title	<Enter Title>		
Cloud Service Provider	CSP Name		

Name	<Enter Name>	Date	<Select Date>
Title	<Enter Title>		
Cloud Service Provider	CSP Name		

FedRAMP 0100011001000101010001000101001001000001010011010101000001001111010 | ∞

Controlled Unclassified Information

4.7. Document Approvals

The OSCAL syntax is the same for document approvers in the SSP, SAP, and SAR. For the SSP, approvers are typically executives within the CSP. For the SAP and SAR, approvers are typically executives within the assessor's organization.

Representation

```
<metadata>
   <!-- title, published ... prop, link -->
   <role id="content-approver">
      <title>[SSP, SAP, or SAR] Approval</title>
      <desc>The executive(s)accountable for the accuracy of this content.</desc>
   </role>
   <role id="cloud-service-provider">
      <title>Cloud Service Provider</title>
      <short-name>CSP</short-name>
   </role>
   <party uuid="uuid-of-csp" type="organization">
      <party-name>Cloud Service Provider (CSP) Name</party-name>
      <short-name>CSP Acronym/Short Name</short-name>
   </party>
   <party uuid="uuid-of-person-1" type="person">
      <party-name>[SAMPLE]Person Name 1</party-name>
      <prop name="title" ns="https://fedramp.gov/ns/oscal">Individual's Title</prop>
      <member-of-organization>uuid-of-csp-party</member-of-organization>
   </party>
   <party uuid="uuid-of-person-2" type="person">
      <party-name>[SAMPLE]Person Name 2</party-name>
      <prop name="title" ns="https://fedramp.gov/ns/oscal">Individual's Title</prop>
      <member-of-organization>uuid-of-csp</member-of-organization>
   </party>
   <responsible-party role-id="cloud-service-provider">
      <party-uuid>uuid-of-csp</party-uuid>
   </responsible-party>
   <responsible-party role-id="content-approver">
      <party-uuid>uuid-of-person-1</party-uuid>
      <party-uuid>uuid-of-person-2</party-uuid>
   </responsible-party>
</metadata>
```

FedRAMP Extension (Person's Title)

prop (`ns="https://fedramp.gov/ns/oscal"`):

- `name="title"`

Defined Identifiers

Required Role IDs:

- `content-approver`
- `cloud-service-provider`

XPath Queries

```
Approver's Name:
  (/*/metadata/party[@uuid=[/*/metadata/responsible-party[@role-id='content-
  approver']/party-uuid]]/party-name)[1]

Approver's Title:
  (/*/metadata/party[@uuid=[/*/metadata/responsible-party[@role-id='content-approver']
  /party-uuid]]/prop[@name='title'][@ns='https://fedramp.gov/ns/oscal'])[1]

NOTE: For each additional approver, replace the "[1]" with "[2]", "[3]", and so on.

CSP Name:
  /*/metadata/party[@uuid=[/*/metadata/responsible-party[@role-id='cloud-service-
  provider']/party-uuid]]/party-name
```

NOTES:

- A person's title is not a valid OSCAL field. A FedRAMP extension must be used to capture a person's title.
- The code above is an SSP example For SAP and SAR, a similar approach is used for the assessor, using the "`assessor`" `role` ID instead of the "`cloud-service-provider`" `role` ID.

12. LAWS, REGULATIONS, STANDARDS AND GUIDANCE

12.1. Applicable Laws and Regulations

The FedRAMP Laws and Regulations can be found on this web page: Templates.

Table 12-1 Information System Name Laws and Regulations includes additional laws and regulations specific to Information System Name.

12.2. Applicable Standards and Guidance

The FedRAMP Standards and Guidance be found on this web page: Templates

Table 12-2 Information System Name Standards and Guidance includes in this section any additional standards and guidance specific to Information System Name.

14. ACRONYMS

The master list of FedRAMP acronym and glossary definitions for all FedRAMP templates is available on the FedRAMP website Documents page.

Please send suggestions about corrections, additions, or deletions to info@fedramp.gov.

4.8. FedRAMP Standard Attachments (Acronyms, Laws/Regulations)

The FedRAMP MS Word-based SSP, SAP and SAR templates included links to the FedRAMP Laws and Regulations file, as well as the FedRAMP Acronyms file posted on https://fedramp.gov.

These are already included in the OSCAL-based FedRAMP templates as resources. The `resource` linking to the FedRAMP citations file is identified with the conformity tag, "`fedramp-citations`". The `resource` linking to the FedRAMP acronyms file is identified with the conformity tag, "`fedramp-acronyms`".

Representation

```
<back-matter>

   <resource uuid="3a5ca2de-0f66-47e6-844d-6ccdf214b767">
      <title>FedRAMP Applicable Laws and Regulations</title>
      <prop name="conformity" ns="https://fedramp.gov/ns/oscal">fedramp-citations</prop>
      <rlink href="https://-cut-/SSP-A12-FedRAMP-Laws-and-Regulations-Template.xlsx" />
   </resource>

   <resource uuid="12da89ef-51dd-4404-948d-e9f0e25b961e">
      <title>FedRAMP Master Acronym and Glossary</title>
      <prop name="conformity" ns="https://fedramp.gov/ns/oscal">fedramp-acronyms</prop>
      <rlink href="https://-cut-/FedRAMP_Master_Acronym_and_Glossary.pdf" />
   </resource>

</back-matter>
```

FedRAMP-Conformity Tags

FedRAMP Standard Citations:

- `fedramp-citations`

FedRAMP Acronyms and Glossary:

- `fedramp-acronyms`

XPath Queries

```
Link to FedRAMP Applicable Laws and Regulations:
  /*/back-matter/resource/prop[@name='conformity'][@ns='https://fedramp.gov/ns/oscal']
  [string(.)='fedramp-citations']/../rlink/@href

Link to FedRAMP Acronyms and Glossary:
  /*/back-matter/resource/prop[@name='conformity'][@ns='https://fedramp.gov/ns/oscal']
  [string(.)='fedramp-acronyms']/../rlink/@href
```

FEDRAMP SYSTEM SECURITY PLAN (SSP) _____ BASELINE TEMPLATE

CSP Name | Information System Name *Version #.#, Date*

12. LAWS, REGULATIONS, STANDARDS AND GUIDANCE

12.1. Applicable Laws and Regulations

The FedRAMP Laws and Regulations can be found on this web page: Templates.

Table 12-1. Information System Name Laws and Regulations includes additional laws and regulations specific to Information System Name.

Instruction: The information system name is a repeatable field that is populated when the Title Page is completed. If the CSP does not have additional laws and regulations that it must follow, please specify "N/A" in the table.

Delete this and all other instructions from your final version of this document.

Table 12-1. Information System Name Laws and Regulations

Identification Number	Title	Date	Link
<Reference ID>	<Reference Title>	<Ref Date>	<Reference Link>
<Reference ID>	<Reference Title>	<Ref Date>	<Reference Link>
<Reference ID>	<Reference Title>	<Ref Date>	<Reference Link>

12.2. Applicable Standards and Guidance

The FedRAMP Standards and Guidance be found on this web page: Templates

Table 12-2. Information System Name Standards and Guidance includes in this section any additional standards and guidance specific to Information System Name.

Instruction: The information system name is a repeatable field that is populated when the Title Page is completed. If the CSP does not have additional standards or guidance that it must follow, please specify "N/A" in the table.

Delete this and all other instructions from your final version of this document.

Table 12-2. Information System Name Standards and Guidance

Identification Number	Title	Date	Link
<Reference ID>	<Reference Title>	<Ref Date>	<Reference Link>
<Reference ID>	<Reference Title>	<Ref Date>	<Reference Link>
<Reference ID>	<Reference Title>	<Ref Date>	<Reference Link>

FedRAMP Extensions and Accepted Values

prop (`ns="https://fedramp.gov/ns/oscal"`):

- `name="type"`
  - **Valid:** `law, regulation, standard, guidance, pii`
- `name="publication"`

4.9. Additional Laws, Regulations, Standards or Guidance

Additional citations must be represented as additional `resource` assemblies. One `resource` assembly per citation. This applies to applicable laws, regulations, standards, or guidance beyond FedRAMP's predefined set.

Each must have a type defined. The value of the type filed must be set to "law", "regulation", "standard", or "guidance" as appropriate. There may be more than one type defined. FedRAMP tools use the type field to differentiate these resource assemblies from others.

Representation

```
<back-matter>
   <resource uuid="d45612a9-cf25-4ef6-b2dd-69e38ba2967a">
      <title>[SAMPLE]Name or Title of Cited Law</title>
      <prop name="type" ns="https://fedramp.gov/ns/oscal">law</prop>
      <prop name="publication" ns="https://fedramp.gov/ns/oscal">Document Date</prop>
      <prop name="version" ns="https://fedramp.gov/ns/oscal">Document Version</prop>
      <doc-id type="doi">Identification Number</doc-id>
      <rlink href="https://domain.example/path/to/document.pdf" />
   </resource>
   <resource uuid="a8a0cc81-800f-479f-93d3-8b8743d9b98d">
      <title>[SAMPLE]Name or Title of Privacy-Related Law Citation</title>
      <prop name="type" ns="https://fedramp.gov/ns/oscal">law</prop>
      <prop name="type" ns="https://fedramp.gov/ns/oscal">pii</prop>
      <prop name="publication" ns="https://fedramp.gov/ns/oscal">Document Date</prop>
      <prop name="version" ns="https://fedramp.gov/ns/oscal">Document Version</prop>
      <doc-id type="doi">Identification Number</doc-id>
      <rlink href="https://domain.example/path/to/document.pdf" />
   </resource>
   <!-- repeat citation assembly for each law, regulation, standard or guidance -->
   <!-- resource -->
</back-matter>
```

XPath Queries

Replace "[1]" with "[2]", "[3]", etc.

```
Number of Laws and Regulations (nteger):
  count(/*/back-matter/resource/prop[@name="type"][@ns="https://fedramp.gov/ns/oscal"]
  [(string(.) = "law") or (string(.)="regulation")])

Laws and Regulations - Identification Number:
  (/*/back-matter/resource/prop[@name="type"][@ns="https://fedramp.gov/ns/oscal"]
  [(string(.) = "law") or (string(.)="regulation")])[1]/../doc-id

Laws and Regulations - Title:
  (/*/back-matter/resource/prop[@name="type"][@ns="https://fedramp.gov/ns/oscal"]
  [(string(.) = "law") or (string(.)="regulation")])[1]/../title

Laws and Regulations - Date:
  (/*/back-matter/resource/prop[@name="type"][@ns="https://fedramp.gov/ns/oscal"]
  [(string(.) = "law") or (string(.)="regulation")])[1]/
  ../prop[@name="publication"][@ns="https://fedramp.gov/ns/oscal"]

Laws and Regulations - Link:
  (/*/back-matter/resource/prop[@name="type"][@ns="https://fedramp.gov/ns/oscal"]
  [(string(.) = "law") or (string(.)="regulation")])[1]/../rlink/@href

NOTE: For Standards and Guidance replace "law" with "standard" and "regulation" with
  "guidance" in the above queries to generate the Standards and Guidance tables.
```

IMPORTANT: At time of publication, `type`, `publication`, and `version` are FedRAMP extensions; however, NIST is evaluating these for inclusion in the core OSCAL syntax. If adopted by NIST, the @ns flag will need to be dropped from the above syntax and XPath queries.

FEDRAMP SYSTEM SECURITY PLAN (SSP) _____ BASELINE TEMPLATE

CSP Name | Information System Name *Version #.#, Date*

15. ATTACHMENTS

A recommended attachment file naming convention is <information system abbreviation> <attachment number> <document abbreviation> <version number> (for example, "Information System Abbreviation A8 IRP v1.0"). Use this convention to generate names for the attachments. Enter the appropriate file names and file extensions in Table 15-1 to describe the attachments provided. Make only the following additions/changes to Table 15-1:

- The first item, Information Security Policies and Procedures (ISPP), may be fulfilled by multiple documents. If that is the case, add lines to Table 15-1. to differentiate between them using the "xx" portion of the File Name. ***Example*** Enter Information System Abbreviation ***A1 ISPP xx v1.0***. Delete the "xx" if there is only one document.
- Enter the file extension for each attachment.
- Do not change the Version Number in the File Name in Table 15-1. . (Information System Abbreviation, attachment number, document abbreviation, version number)

Table 15-1. Names of Provided Attachments

Attachment	File Name	File Extension
Information Security Policies and Procedures	Enter Information System Abbreviation A1 ISPP xx v1.0	. enter extension
User Guide	Enter Informa	
Digital Identity Worksheet	Included in Se	
PTA	Included in Se	
PIA (if needed)	Enter Informa	
Rules of Behavior	Enter Informa	
Information System Contingency Plan	Enter Informa	
Configuration Management Plan	Enter Informa	
Incident Response Plan	Enter Informa	
CIS Workbook	Enter Informa Workbook v1.	
FIPS 199	Included in Section 15	
Inventory		

FedRAMP-Conformity Tags

FedRAMP Privacy Impact Assessment:

- `privacy-impact-assessment`

IMPORTANT

Most required attachments to the SSP, SAP and SAR either have a direct OSCAL syntax representation, or appear as a `resource` with a conformity tag.

See the FedRAMP OSCAL Registry for a complete list of Conformity Tags in back-matter

Conformity Tags are only used where OSCAL syntax alone is not specific enough for FedRAMP tools to clearly identify required content. For example, in an SSP, the authorization boundary diagram in back-matter should always be referenced by the authorization-boundary assembly, making its purpose clear. So no conformity tag is used. Other attachments must have a conformity tag to allow for different IDs and titles, yet ensure FedRAMP tools know which attachment satisfies each requirement.

See the FedRAMP OSCAL Registry for a complete list of conformity tags, and look for content-specific lists in each Guide.

4.10. Attachments and Embedded Content

There are several attachments in a classic FedRAMP MS Word based SSP, SAP, SAR document or Deviation Request (DR) form. Some lend well to machine-readable format, while others do not. Those that are readily modeled in machine-readable format are typically addressed within the OSCAL syntax, while attachments such as policies, procedures, plans, guides, and rules of behavior documents are all treated as attachments in OSCAL as well.

Further, any diagrams or images that normally appear in context, such as the authorization boundary diagram, are attached in the back-matter and referenced from the body of the OSCAL file, as described in Section 2.7 *Citations, Attachments, and Embedded Content in OSCAL Files*. The following table represents attachments and embedded content.

FedRAMP Extensions and Accepted Values

prop (`ns="https://fedramp.gov/ns/oscal"`):

- `name="type"`
  - **Valid:** `policy, procedure, guide, pia, rob, plan`
- `name="publication"`
- `name="version"`

Attachment Representation

```
<!-- cut -->
<back-matter>
  <!-- citation -->
      <resource uuid="fab59751-b855-40cb-93c1-492562e20e18">
    <title>Privacy Impact Assessment</title>
    <prop name="conformity" ns="https://fedramp.gov/ns/oscal">privacy-impact-assessment</prop>
    <prop name="publication" ns="https://fedramp.gov/ns/oscal">Document Date</prop>
    <prop name="version" ns="https://fedramp.gov/ns/oscal">Document Version</prop>
    <!-- Add rlink with relative path OR embed with base64 encoding -->
    <rlink href="./pia.docx" />
    <base64>00000000</base64>
  </resource>

  <resource id="diag-boundary-1">
    <desc>The primary authorization boundary diagram.</desc>
    <!-- Add rlink with relative path or embed with base64 encoding -->
    <rlink href="./boundary.png" />
    <base64>00000000</base64>
    <remarks><p>Set system-characteristics/authorization-boundary/diagram/link/@href =
                "#diag-boundary-1"</p></remarks>
  </resource>
</back-matter>
```

Replace "policy" with "plan", "rob", etc. for each attachment type.

XPath Queries

```
PIA Attachment (Embedded Base64 encoded):
  /*/back-matter/resource/prop[@name='conformity'][@ns='https://fedramp.gov/ns/oscal']
  [string(.)='privacy-impact-assessment']/.. /base64

PIA Attachment (Relative Link):
  /*/back-matter/resource/prop[@name='conformity'][@ns='https://fedramp.gov/ns/oscal']
  [string(.)='privacy-impact-assessment']/.. /rlink/@href

Publication Date of PIA:
  /*/back-matter/resource/prop[@name='conformity'][@ns='https://fedramp.gov/ns/oscal']
  [string(.)='privacy-impact-assessment']/../prop[@name="publication"]
  [@ns="https://fedramp.gov/ns/oscal"]
```

Tools creating OSCAL content should include a `media-type` for all `rlink` and `base64` fields, as well as a `filename` for all `base64` fields.

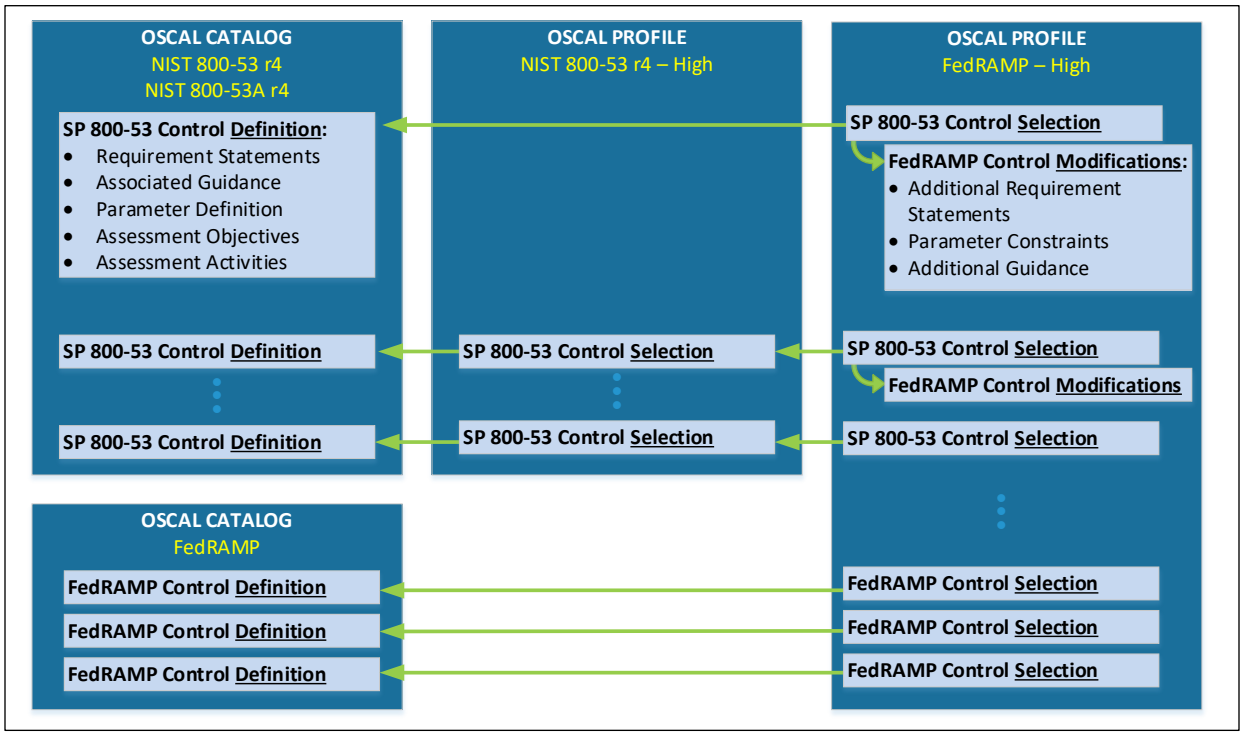
Tools processing rlink and base64 content with or without these fields. Where present they should be used when validating or rendering the linked or embedded content.

APPENDICIES

APPENDIX A. OSCAL-BASED FEDRAMP BASELINES

NIST designed OSCAL catalogs as the primary source of control definition information from a framework publisher. Catalogs are typically only published by organizations such as NIST for NIST SP 800-53, or ISO/IEC for standards such as their 27000 series. If an organization has unique control definitions that fall outside an applicable framework, the organization must create a catalog containing those unique control definitions.

NIST designed profiles as the primary means of defining a baseline of controls. An OSCAL profile may identify and even modify controls from one or more catalogs and even from other profiles. This approach ensures control additions, modifications, or removal are fully traceable back to the source of the modification.



FedRAMP's baselines are represented as OSCAL profiles. The correct profile must be selected from the SSP based on the system's identified security categorization level. This can be checked using the XPath syntax below.

Security Sensitivity Level XPath Query

```
(SSP) Security Categorization Level:
  /*/system-characteristics/security-sensitivity-level
```

This determines which URL should be entered for the `import-profile` field in an OSCAL-based FedRAMP SSP.

Baseline Representation

```
    <!-- metadata -->
    <!-- This must point to the appropriate FedRAMP Baseline -->
    <import-profile href="https://path/to/FedRAMP_MODERATE-baseline_profile.xml"/>
    <!-- system-characteristics -->
```

FedRAMP validation tools will compare the identified security categorization level to the actual FedRAMP baseline specified in the SSP and raise a warning if a different baseline was used.

High

XML Version:
https://raw.githubusercontent.com/usnistgov/OSCAL/master/content/fedramp.gov/xml/FedRAMP_HIGH-baseline_profile.xml

JSON Version:
https://raw.githubusercontent.com/usnistgov/OSCAL/master/content/fedramp.gov/json/FedRAMP_HIGH-baseline_profile.json

Moderate

XML Version:
https://raw.githubusercontent.com/usnistgov/OSCAL/master/content/fedramp.gov/xml/FedRAMP_MODERATE-baseline_profile.xml

JSON Version:
https://raw.githubusercontent.com/usnistgov/OSCAL/master/content/fedramp.gov/json/FedRAMP_MODERATE-baseline_profile.json

Low

XML Version:
https://raw.githubusercontent.com/usnistgov/OSCAL/master/content/fedramp.gov/xml/FedRAMP_LOW-baseline_profile.xml

JSON Version:
https://raw.githubusercontent.com/usnistgov/OSCAL/master/content/fedramp.gov/json/FedRAMP_LOW-baseline_profile.json

Do not copy and modify the FedRAMP baseline. FedRAMP will use the original, published file for validation, ignoring any modified copies.

If you require a modification to the FedRAMP baselines, such as may be required when directed to do so by an authorizing official, first contact FedRAMP to coordinate the modification, then follow the instructions in Appendix B.

FedRAMP Tailored

FedRAMP Tailored for Low Impact – Software as a Service (LI-SaaS) Appendix B merges SSP, SAP, and SAR information into a single document. The SSP portions of that document may be represented using the same OSCAL conventions described in this document with only a few minor differences.

Specific OSCAL-based FedRAMP Tailored guidance will be published at a later date; however, fully representing Appendix B in OSCAL requires the SSP, SAP, and SAR syntax, used the same way as they are explained for FedRAMP Low, Moderate, and High baselines.

For your convenience, FedRAMP has made the FedRAMP Tailored for LI-SaaS baseline available now in both XML and JSON formats as follows:

Low-Impact SaaS (Tailored)
XML Version: https://raw.githubusercontent.com/usnistgov/OSCAL/master/content/fedramp.gov/xml/FedRAMP_LI-SaaS-baseline_profile.xml **JSON Version:** https://raw.githubusercontent.com/usnistgov/OSCAL/master/content/fedramp.gov/json/FedRAMP_LI-SaaS-baseline_profile.json

APPENDIX B. MODIFYING A FEDRAMP BASELINE

OSCAL is designed to allow modification of controls and baselines, while maintaining traceability through each layer of modification. This means you must create a new profile as a means of modifying an existing profile.

If you require a change to a FedRAMP baseline, you should first coordinate that change with the FedRAMP JAB or PMO. Assuming FedRAMP agrees with the change, the correct way to implement the change is as follows:

Create a new profile, importing to the appropriate FedRAMP profile, then use profile syntax to make necessary changes.

FedRAMP does not typically allow modifications to its baselines. This capability is present only in the event of a policy change or unforeseen circumstances.

1. **Create a new, blank OSCAL Profile.**
2. **Point to the FedRAMP Baseline**: Point it to the appropriate FedRAMP baseline using an `import` field.
3. **Select the Relevant Controls**: Use the `include` and `exclude` fields to identify the controls to include or exclude from the FedRAMP baseline.
   a. For example, if you need all but one control, you can `include all`, then `exclude` the one.
4. **Specify How Controls Are Organized**: FedRAMP prefers you merge "as-is" using those merge fields. This is relevant when resolving the profile. See the *Profile Resolution* section of this appendix for more information.
5. **Modify the Selected Controls**: Use the `modify` assembly to make modifications to parameters and control definitions.

The next page contains an example profile, which accomplishes the following actions:

- Imports the FedRAMP Moderate baseline
- Includes all controls from that baseline
- Explicitly removes AT-4 from the baseline
- Indicates that if this profile is resolved, the organization of the controls should remain as-is. See the *Profile Resolution* section of this appendix for more information.
- Adds a constraint to the third parameter of AC-1 (ac-1_prm_3), which is more restrictive than the FedRAMP constraint, but changing it from "at least annually" to "at least every six months."
- Removes the additional FedRAMP requirement statement in AU-11 and replaces it with a more restrictive statement, which now requires online retention of audit records for at least 180 days instead of 90 days.

For more information on working with profiles, please visit the NIST OSCAL site at:
https://pages.nist.gov/OSCAL

A complete OSCAL profile syntax reference is available here:
https://pages.nist.gov/OSCAL/documentation/schema/profile/

Sample Profile to Modify a FedRAMP Baseline

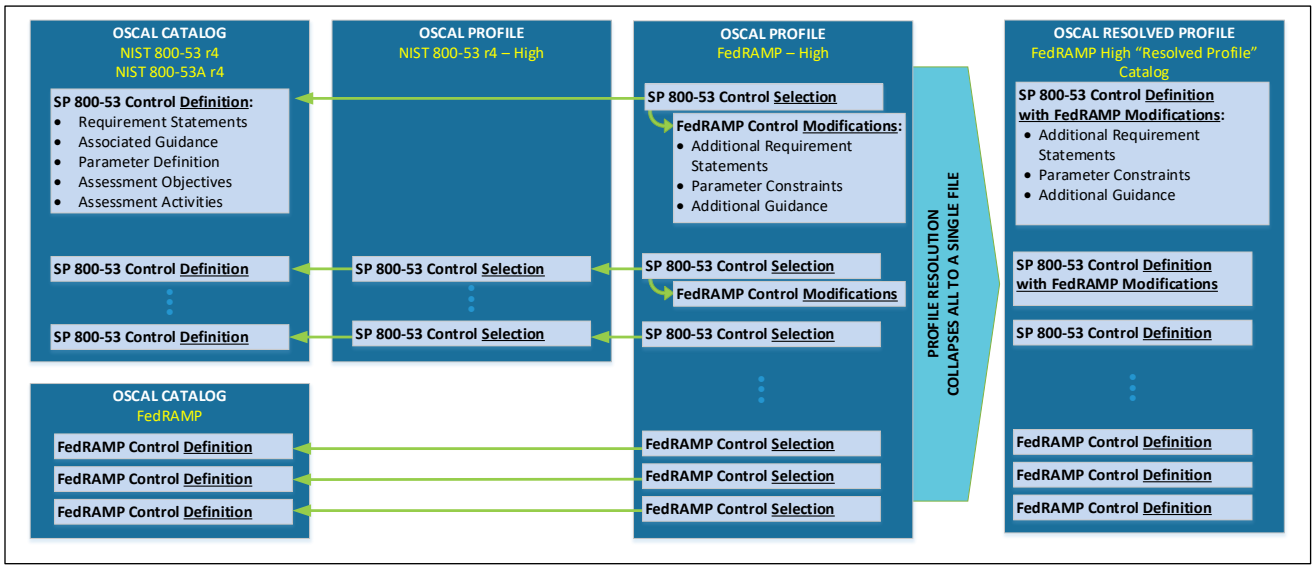
```
<profile xmlns="http://csrc.nist.gov/ns/oscal/1.0"
      uuid="-UUID-value-cut-">
      <metadata>
            <title>[XYZ Org] Modification to FedRAMP Moderate
Baseline</title>
            <last-modified>2019-10-01T11:03:27.392-04:00</last-modified>
            <version>1.1</version>
            <oscal-version>1.0.0-milestone2</oscal-version>
      </metadata>
      <import href="https://path/to/FedRAMP_MODERATE-baseline_profile.xml">
            <include>
                  <!-- Include every control in the Moderate baseline -->
                  <all with-child-controls="yes" />
            </include>
            <exclude>
                  <!-- Remove Control AT-4 -->
                  <call control-id="at-4" />
            </exclude>
      </import>
      <merge><as-is>yes</as-is></merge>
      <modify>
            <set param-id="ac-1_prm_3">
                  <!-- Change the constraint from "at least annually" -->
                  <constraint>at least every six months</constraint>
            </set>
            <remove id-ref="au-11_fr" />
            <alter control-id="au-11">
                  <add position="ending">
                        <part id="au-11_fr" name="item">
                              <title>[XYZ Org]Modified Requirement</title>
                              <part id="au-11_fr_smt.1" name="item">
                                    <prop name="label">Requirement:</prop>
                                    <p>The service provider retains audit
records on-line for at 180 days and further preserves audit records off-line
for a period that is in accordance with NARA requirements.</p>
                              </part>
                        </part>
                  </add>
            </alter>
      </modify>
</profile>
```

APPENDIX C. PROFILE RESOLUTION

Profiles are intended to identify upstream sources of control definition information and show only the changes to those upstream sources. This enables humans and computers to trace control definition changes back to their source framework.

Profile resolution flattens or merges a profile and its imported catalog(s) and profiles into a single OSCAL file using the catalog syntax.

While this ensures traceability of selected controls and modified content, it can also be resource intensive. Profile resolution flattens or merges a profile and its imported catalog(s) and profiles into a single OSCAL file using the catalog syntax.



This single file is essentially a pre-processed result of the profile import and modification content. A resolved profile catalog is useful for the FedRAMP baselines given their static nature. Any tool that would normally open an OSCAL-based FedRAMP profile and process it against the NIST SP 800-53 catalog can instead simply use the resolved-profile catalog.

Each FedRAMP XML and JSON baseline profile has a resolved profile catalog in the same location as the pre-processed profile. Where available, these may be used by tools to save processing time.

The `merge` assembly within an OSCAL profile offers a profile resolver control over how the final file is organized. To maintain the same organization as within the catalog, simply use the `as-is` field and set it to "yes."

The complete profile syntax is available here:

https://pages.nist.gov/OSCAL/documentation/schema/profile-layer/profile/

NIST's Profile Resolution Tool

NIST created a DRAFT profile resolution specification, and built a profile resolution capability based on that specification. The specification can be found here:

https://github.com/usnistgov/OSCAL/tree/master/src/specifications/profile-resolution

The actual tool can be found here:

https://github.com/usnistgov/OSCAL/tree/master/src/utils/util/resolver-pipeline

Currently the tool requires the profile and all imported catalogs and profiles to be in XML format. For now JSON content must be converted to XML before using this tool.

The tool is requires an XSLT 3.1 processor, which is the same requirement for XML to JSON and JSON to XML conversions.

All XSL files provided at the link above must be in the same directory. Only oscal-profile-RESOLVE.xsl must be identified to the XSLT processor. All other files are called by this file as part of processing.

It is also possible to run the scripts directly from the NIST OSCAL repository by supplying the following URL directly to the XSLT processor:

https://raw.githubusercontent.com/usnistgov/OSCAL/master/src/utils/util/resolver-pipeline/oscal-profile-RESOLVE.xsl

This is a new capability provided by NIST and leveraged by FedRAMP. Please report bugs or provide feedback related to this tool directly to NIST at oscal@nist.gov or by submitting an issue here:

https://github.com/usnistgov/OSCAL/issues

APPENDIX D. WORKING WITH ROLES, LOCATIONS, PEOPLE, AND ORGANIZATIONS

An OSCAL file defines roles, people, and organizations within the metadata as part of three separate assemblies:

- **`role`**: A role ID and role title are required. Other content, such as a short-name, description, or remarks are optional.
- **`location`**: Locations, such as corporate offices and data center addresses, are defined as `location` assemblies
- **`party`**: People and organizations are defined as `party` assemblies. An organization is any collection of people, and can represent a company, agency, department, or team.
- **`responsible-party`**: Links roles to parties. The same role can have more than one party assigned to it. Also a party can be assigned to more than one role.

FedRAMP Defined Party Identifiers

FedRAMP has eliminated the use of FedRAMP-Defined Party Identifiers.

With the transition from ID to UUID for party identifiers this is no longer possible. Further, this helps ensure OSCAL remains compatible with multiple compliance frameworks.

Working with Role Assemblies

All roles within the document are defined under the metadata element as follows:

```
<metadata>
   --- cut ---
   <role id="prepared-by">
      <title>Prepared By</title>
      <desc>The organization that prepared this SSP.</desc>
   </role>
   <role id="prepared-for">
      <title>Prepared For</title>
      <desc>The organization for which this SSP was prepared</desc>
   </role>
   --- cut ---
</metadata>
```

To ensure consistent processing, FedRAMP has defined a specific set of roles that must exist with a FedRAMP SSP, SAP, SAR, or POA&M. **Most are pre-populated in the OSCAL-based FedRAMP Templates.** They vary by template.

OSCAL-based FedRAMP tools must ensure these roles, titles, and descriptions exist using the prescribed role IDs within an OSCAL-based FedRAMP file. Additional roles may be added, provided these roles remain.

NIST-defined and FedRAMP-defined role-identifiers are cited in relevant portions of each guide, and summarized in the FedRAMP OSCAL Registry.

Working with Location Assemblies

The location assembly is intended to represent the address of a location such as the HQ of a CSP or 3PAO, a data center, or an Agency.

Some locations are required. For example, the SSP must contain the at least one `location` assembly with the primary business address of the CSP. The SAP and SAR must contain at least one `location` assembly with the primary business address of the assessor.

OSCAL allows the title `field` to be optional. FedRAMP strongly encourages its use. If the country field is missing, FedRAMP tools will assume the address is within the United States of America.

```
<metadata>
   <!-- role -->
   <location uuid="uuid-of-hq-location">
      <title>CSP HQ</title>
      <address type="work">
         <addr-line>1234 Some Street</addr-line>
         <city>Haven</city>
         <state>ME</state>
         <postal-code>00000</postal-code>
         <country>us</country>
      </address>
   </location>
   <!-- party -->
</metadata>
```

Some locations require conformity tags to ensure FedRAMP tools can accurately identify required content. For example, the location assembly for a data center must include a "data-center" conformity tag. For more information, see *Section 3.1.2, FedRAMP Conformity Tagging*.

FedRAMP conformity tags are cited in relevant portions of each guide, and summarized in the FedRAMP OSCAL Registry.

Working with Party Assemblies

Party assemblies may be used to represent individuals, teams, or an entire company/agency. When representing an individual, the `type` flag must have a value of "`person`". When representing a team, company or agency, the `type` flag must have a value of "`organization`".

FedRAMP artifacts typically require an individual's title to be identified; however, OSCAL does not currently provide a field for this. As a result, a FedRAMP extension has been added and must be used for any individual filling a role where FedRAMP requires the individual's title, such as for the system owner.

Contact details, such as an individual's email address and phone number, or a business web site, may be included and are often required within FedRAMP artifacts. A short-name filed provides an ability to define an organization's acronym or desired abbreviation. This is required for the CSP, assessor, and any Agency.

```
<metadata>
   <!-- role -->
   <party uuid="uuid-of-csp" type="organization">
      <party-name>Cloud Service Provider (CSP) Name</party-name>
      <short-name>CSP Acronym</short-name>
      <link href="https://www.csp.com" />
   </party>

   <party uuid="uuid-of-person-1" type="person">
      <party-name>[SAMPLE]Person Name 1</party-name>
      <prop name="title"
            ns="https://fedramp.gov/ns/oscal">Individual's Title</prop>
      <email>name@org.domain</email>
      <phone>202-000-0000</phone>
   </party>
</metadata>
```

FedRAMP extensions are cited in relevant portions of each guide, and summarized in the FedRAMP OSCAL Registry.

Identifying Organizational Membership of Individuals

An individual may be affiliated with one or more teams/organizations.
Use one `member-of-organization` field for each team, and one to link the individual to their employer.

```
<metadata>
   <!-- role -->
   <party uuid="uuid-of-csp" type="organization">
      <party-name>Cloud Service Provider (CSP) Name</party-name>
   </party>

   <party uuid="uuid-of-it-dept type="organization">
      <party-name>CSP's IT Department</party-name>
   </party>

   <party uuid="uuid-of-person-1" type="person">
      <party-name>[SAMPLE]Person Name 1</party-name>
      <member-of-organization>uuid-of-csp</member-of-organization>
      <member-of-organization>uuid-of-it-dept</member-of-organization>
   </party>
</metadata>
```

Identifying the Address of People and Organizations

Representing the address of people or organizations (parties) may be accomplished with one of three approaches:

Preferred Approach: When multiple parties share the same address, such as multiple staff members at a company HQ, define the address once as a `location` assembly, then use the `location-uuid` field within each `party` assembly to identify the location of that individual or team.

Alternate Approach: If the address is unique to this individual, it may be included in the `party` assembly itself.

Hybrid Approach: It is possible to include both a `location-uuid` and an `address` assembly within a `party` assembly. This may be used where multiple staff are in the same building, yet have different office numbers or mail stops. Use the `location-uuid` to identify the shared building, and only include a single `addr-line` field within the party's `address` assembly.

A tool developer may elect to always create a location assembly, even when only used once; however, tools must recognize and handle all of the approaches above when processing OSCAL files.

```
<metadata>
   <!-- cut -->
   <location uuid="uuid-of-hq-location">
      <title>CSP HQ</title>
      <address type="work">
         <addr-line>1234 Some Street</addr-line>
         <city>Haven</city>
         <state>ME</state>
         <postal-code>00000</postal-code>
      </address>
   </location>

   <party uuid="uuid-of-csp" type="organization">
      <party-name>Cloud Service Provider (CSP) Name</party-name>
      <location-uuid>uuid-of-hq-location</location-uuid>
   </party>

   <party uuid="uuid-of-person-1" type="person">
      <party-name>[SAMPLE]Person Name 1</party-name>
      <address>
         <addr-line>Mailstop A-1</addr-line>
      </address>
      <location-uuid>uuid-of-hq-location</location-uuid>
   </party>
</metadata>
```

Working with Responsible Party Assemblies

The responsible party assembly links party assemblies (people and organizations) to defined roles. FedRAMP tools rely on these linkages to find required content.

For example, an OSCAL-based SSP must have a role defined for the System Owner using the role ID value "system-owner". The responsible-party assembly links this required role to the individual (party). FedRAMP tools follow this linkage to verify that a system owner was identified in the SSP, and to display that information to reviewers.

```
<metadata>
   <!-- party -->
   <responsible-party role-id="system-owner">
      <party-uuid>uuid-of-person-1</party-uuid>
   </responsible-party>
</metadata>
```