Overview

This includes overview topics of the OSCAL Foundation Patterns Library

Welcome

The goal of the OSCAL Patterns Library is to maximize interoperability across OSCAL tools. The library accomplishes this by defining the recommended OSCAL representation for specific use cases. Recommendations are based on the consensus of participating Foundation members.

Adoption

The OSCAL Foundation has defined the following adoption strategies:

- FedRAMP SSP Adoption Strategies
- Prioritization of additional artifacts TBD

Leave Comments

If you have feedback on any of the content, please leave a comment on the page. Commenting is enabled after you self-register using the "Log in" button in the upper right.

Library Organization

- **Core OSCAL**: Patterns, guidance, information and resources common to any OSCAL representation regardless of framework or control set.
- **FedRAMP**: Patterns, guidance, information and resources for expressing FedRAMP Authorization Packages in OSCAL.
  - **FedRAMP System Security Plan (SSP)**: Represent FedRAMP SSP content in OSCAL (***In Progress***)
  - **FedRAMP Plan of Action and Milestones (POA&M)**: Represent FedRAMP POA&M content in OSCAL (**Next**)
  - **FedRAMP Assessments**: Represent FedRAMP SAP and SAR content in OSCAL (**Future**)
- **Additional Frameworks and Industries**: *Additional frameworks and industries are prioritized based on demand and available resources.*)

Project Status and Next Steps

The OSCAL Foundation jumpstarted this library using prior content created by former FedRAMP PMO members. The initial deployment focuses on deployment and cleanup of that FedRAMP-specific content in response to new OSCAL requirements for FedRAMP-

authorized systems.

See the Milestones, Approach and Status for completed work, current status and future steps.

Governance

Content in this library is drafted by Technical Focus Groups (TFG). Each TFG reports status at the weekly Technology Working Group (TWG) meeting, held each Tuesday at 11:00 AM Eastern Time.

All content is subject to potential review and approval by the Technical Advisory Group (TAG). Steering Committee and/or Board. approval is also required for more changes to core OSCAL, strategic or broad-reaching topics; and to resolve conflicts.

The organizational structure of the Foundation is as follows:

- Board of Directors
  - Executive Steering Committee
    - Technical Advisory Group (TAG)
      - Technical Working Group (TWG) *
        - Multiple Technical Working Groups (TFG)

* There are other OSCAL Foundation working groups, which are not currently involved in this effort.

Getting Involved

The OSCAL Foundation TWG and TFGs are open to the public. Join the appropriate OSCAL Foundation group lists to see meeting details and receive announcements.

Paid membership is required for approval or voting rights, and to ensure the Foundation's future viability. See the Get Involved page to enquire about membership.

Milestones, Approach and Status

The OSCAL Foundation's *FedRAMP Technical Focus Group (TFG)* is enabling FedRAMP stakeholders to adopt OSCAL for FedRAMP package deliverables. The following is our plan of work:

Milestones

- **Phase 0**: Form Team and Establish Resources [*Complete*]
- **Phase 1**: FedRAMP System Security Plans (SSP) [*In Progress*]
- **Phase 2**: FedRAMP Plan of Action and Milestones (POA&M) [*Re-evaluating priority in light FedRAMP Notice 9 Item #3*]
- **Phase 3**: FedRAMP Security Assessment Plans and Reports (SAP and SAR)
- **Phase 4**: Advanced and Refinement
- **Phase 5**: FedRAMP Adjacent Frameworks (GovRAMP, DoD/FedRAMP+, DoD Impact Levels, CMMC and Related Variants)
- **Future**:: Other Frameworks (PCI CSA, CIS, DSS, SOC 2, ISO-270xx, etc.)

Target Dates

- March 31: Full Draft SSP
- April: Socialize with FedRAMP PMO and CSP-AB
- April 15: Presentation at NIST OSCAL Workshop

Approach

Work within each of the above phases occurs in this sequence:

1. **Define the OSCAL MVP Representation**
2. **Address Validation**:
3. **Communicate Availability**
4. **Expand and Refine Representation**

Status Log

Last Updated April 8, 2026

- Form TFG: *Complete*
- Establish Patterns Library: *Complete*
- Establish GitHub Repository: *Complete*
- Migrate prior FedRAMP baselines in OSCAL format to repository: *Complete*
- Migrate prior FedRAMP OSCAL SSP work into patterns library: *Complete*
- Formulate communication plan: *Complete*
- Migrate prior FedRAMP OSCAL SSP example: *Complete*
- Formulate Adoption Paths: *Complete*

- Review/Refine FedRAMP OSCAl SSP patterns: *In Progress*
- Review/Refine FedRAMP OSCAL SSP example: *In Progress*
- Draft "Getting Started" content: *Next*
- POA&M example and patterns: *Next*

Metaschema Authoring Principles

The original OSCAL Technical Team had normalized on several guiding principles for authoring metaschema content that were not captured. Going forward, as topics come up, they will be added here as *suggested* principles and/or to capture historic guiding principles for awareness and discussion.

Nothing in this chapter is authoritative at this time, but could become authoritative in the future if reviewed and approved by an appropriate governing body.

Defining Allowed Values

> This page is still under development.

The `<allowed-values>` assembly provides a consistent and unambiguous list of **machine-readable tokens** to be used as data for an identified OSCAL field or flag values.

Human readability is coincidental and **not** their intended purpose.

Taxonomy

In metaschema, allowed values are a type of constraint applied to a metaschema flag or field. They are applied to the target of the `<constraint>`.

The `<allowed-values>` assembly consists of:

- an optional `@id` flag: A unique identifier within the OSCAL metaschema. Used for error reporting and debugging.
- an optional `@allow-other`: The default value is `no`, which means these values are stictly enforced and no other value is allowed. When `yes` other values are allowed.
- one `<enum>` element for each allowed value, containing:
  - a required `@value` set to the machine-readable token. **These are case sensitive**
  - required *text* content inside the element describing the concept represented by the token.

Example: Party Type Allowed Values

```
NEED DIFFERENT EXAMPLE
```

This example says, "**{{INSERT}}**".

In this example, the constraint's target is implied by its placement within the **{{INSERT}}**. Some constraint targets are explicitly defined using `<target />`.

Guiding Principles

When defining or modifying an allowed values set, the following principles should be applied to the greatest degree practical:

- **Mutually Exclusive**: Each allowed value should be cleanly distinct from any other allowed value in the set. There should be no ambiguity as to which value should be used.
- **Collectively Exhaustive**: Every possible concept should be represented.

When a set of values has been well researched and consensus agrees the set is both *mutually exclusive* and *collectively exhaustive*, the `allow-other` should be set to `no`.

When it is not practical to achieve *collectively exhaustive*, `allow-other` should be set to `yes`. This can happen when a an allowed value set needs to be able to grow to accomodate an evolving industry and when an exhaustive research effort to identify all possible values is not practical due to resource constraints.

Applying the Principles

The above example **{{INSERT}}**, and does not allow other values.

This is mutually exclusive in that it separates **{{INSERT}}**.

They reflect the concepts of **{{INSERT}}**.

It would ...**{{INSERT}}**

Reference

Read more about the Mutually Exclusive / Collectively Exhaustive (MECE) Principle.