Virtual Room #1

Hosted By: Nikita Wootten, Computer Scientist, NIST

**Disclaimer:** Portions of the event may be recorded, and audience Q&A or comments may be captured. The recorded event may be edited and rebroadcasted or otherwise made publicly available by NIST. By attending this event, you acknowledge and consent to having your conversation recorded.
OSCAL "Deep Diff"
- a model-agnostic OSCAL tool and the concept behind it -
How do I track changes that my team has made between revisions of a document?

The Problem: Large Documents are Difficult to Digest

Authors
How do I track changes that my team has made between revisions of a document?

Catalog Consumers
How can I produce a checklist of controls with relevant changes when a new revision of a control catalog comes out?

Developers
How can I track when certain types of changes to a document happens, and make decisions based on those change-lists (such as in a CI/CD pipeline)?
The Solution: A “Diff” Tool for OSCAL Documents

- A tool that can generate a comparison between two OSCAL documents
- Configurable enough to be applied in multiple scenarios
- Must be able to generate output documents that are easy to digest and share
- Portable and extendable so that it can be integrated into other tools (such as web applications)

GitHub’s diff view, an example of a diff tool used daily by developers
An open-source JavaScript/TypeScript CLI application and library that can be used to compare arbitrary JSON documents

- Does not rely on a schema to compare objects, can be configured to compare documents in a reproducible manner
- Generates outputs in multiple formats including easy-to-distribute Excel spreadsheets
- Can be integrated into other tools, including web and desktop applications
Scenario: Comparing two SSPs
By default, OSCAL deep diff produces a JSON document listing the differences between the two documents. Valid change types are “property_left_only”, “property_right_only”, “property_changed”, and “array_changed”. Each “array_changed” type has a sub-list of changes for each matched pair of items.

The raw JSON document can be used to produce friendlier output documents. Excel output collects all of one object type (like controls) and displays them in an Excel document. The tool can be extended to produce other comparison views (such as a web-application or pdf report).
The tool can be configured to change the behavior of the comparison:

- Ignore objects that are irrelevant to the comparison
- Change the way properties are compared (select a string similarity algorithm, ignore case, etc.)
- Swap out the algorithms used to “match” array items to each other

...as well as the output format:

- Change which objects will be collected for the comparison
- Choose which metadata should be displayed in the output document
- Output to JSON, Excel, etc.

This is all configured via a YAML file.
Scenario: Comparing Component Definitions
Scenario: Comparing SP 800-53 Revisions
Shortcomings

• **Speed of comparisons**
  • Array comparison algorithms are computationally expensive.
  • For example, depending on the settings used, comparisons between SP 800-53 revisions can take upwards of 10 minutes.

• **Comparison behavior tuning**
  • Getting the tool fit a particular comparison scenario may require tweaking.
  • This can be solved with community support and examples.

• **Comparison results**
  • Some scenarios are not supported yet, such as object demotion/promotion. (ex. A control becoming an enhancement)
Call to Action

If this tool is exciting or potentially useful to you:

• Please provide feedback, report bugs, and suggest improvements!
• Feel free to submit issues, PRs, and discussions to https://github.com/usnistgov/oscal-deep-diff

Please note: The version of OSCAL Deep Diff shown here is still experimental, see https://github.com/usnistgov/oscal-deep-diff/pull/34
Questions?