OSCAL Mini Workshop

Agency Security Assessment Framework With OSCAL ComplianceOps

Nov 03, 2022
Agency Perspective

Intended Audience

- Agency Officials who want to [re]use cloud services
- Agency staff who perform reviews
- **Agency Devs & SMEs who want to help automate and innovate!**
WARNING: Future Closer than it Appears

DISCLAIMER
Does not represent official NIST, FedRAMP, US Gov or Agency statements, plans, commitments, or directives.

All recommendations, tools for the OSCAL community; opinions solely the author’s.
Find Inconsistencies

- across* SSP control narratives
- between* diagrams ↔ SSP narrative
- between* SSP ↔ 3PAO Test Cases
- between* the SAR ↔ POA&M
- between* ConMon Vulns ↔ Controls ↔ Threats ↔ IR

*More on this later!
Implement Agency (Customer) Responsibilities

CIS/CRM Impact
Assess Control Implementations
Assess Policies and Procedures
Assess 3PAO Activities
Supervise and Review ConMon
Agency FedRAMP Responsibilities

“Reuse” of existing ATO Still requires Agency work

FIPS 199
NIST 800-63B Digital Identity Determination
SSP+SAP
SAR+POA&M
Leveraged ATOs and External Services

Committing to Monthly ConMon Review and Analysis
Risk and Security Control Assessment

Evaluation vs. Testing?
Connected to Threats?
Security Capabilities?
Measure Coverage?
Security Assessment Issues Today

Manual: CTRL+C, CTRL+V to GRC
Control Selection: Ad hoc mapping
Implementation Gaps: ORs, DRs OK?
Inheritance: Changes? Traceable?
3PAO Assessment: Test coverage?
RET Assessment: SME requirements?
POA&M, IR: Inventory $\Delta$, Sec Posture
Annual, SCRs: Here we go again!
Sponsorship Problems Inhibit Innovation

Staff: “We don’t have the staff time to sponsor”
Timeline: “Takes too long”
Budget: “Don’t have resources for new packages”
ConMon: “Don’t have tools to manage this”
FUD? “We tried sponsoring, it was hard”

Adversaries 💖 status quo
No FedRAMP requirement to map vulns to Controls based on Threats because impossible to do manually*

* Don’t do it manually! Surprises below...
FedRAMP’s Stated Goal for Automation

Feb 2021

“Expedite the assessment, adjudication of artifacts”

“Shift LoE away from compliance to risk management”

Reduce This... More importantly... 4-16
How does OSCAL Help?

Clarity
“see” “look at” “not showing” “does not depict” “connecting lines” “ambiguous” “unclear”

vs

1987
{
1988
    "uuid": "7a36cf53-156d-4d1f-9a8b-433f61cc57b7",
1989
    "control-id": "ac-2",
1990
    "props": [
1991
        {
**How does OSCAL Help?**

**Completeness**

“Incomplete* or poor response...that compromises security”

“Pet pees” “try to pull the wool”

*More on this later!

<table>
<thead>
<tr>
<th>Section B: Documents Provided Check</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0 Initial Authorization Package Checklist</td>
</tr>
<tr>
<td>2.0 ATO Provided</td>
</tr>
<tr>
<td>3.0 System Security Plan (SSP)</td>
</tr>
<tr>
<td>3.1 AK: 1. Informative Security Policies and Procedures</td>
</tr>
<tr>
<td>3.2 AK: 2. User Guide</td>
</tr>
<tr>
<td>3.3 AK: 3. Digital Identity Worksheet</td>
</tr>
<tr>
<td>3.4 AK: 4. Risky Threats Analysis (PTA) and Privacy Impact Assessment (PIA)</td>
</tr>
<tr>
<td>3.5 AK: 5. Roles of Behavior (ROB)</td>
</tr>
<tr>
<td>3.6 AK: 6. Information System Contingency Plan (ICP)</td>
</tr>
<tr>
<td>3.7 AK: 7. Configuration Management Plan (CMP)</td>
</tr>
<tr>
<td>3.8 AK: 8. Incident Response Plan (IRP)</td>
</tr>
<tr>
<td>3.9 AK: 9. Control Implementation Summary (CIS) Workbook</td>
</tr>
<tr>
<td>3.11 AK: 11. Separation of Duties Matrix</td>
</tr>
<tr>
<td>3.12 AK: 12. FedRAMP Laws and Regulations</td>
</tr>
<tr>
<td>3.13 AK: 13. FedRAMP Migrated Inventory Workbook</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Section C: Overall SSP Checks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is the correct SSP Template used?</td>
</tr>
<tr>
<td>Is the correct Deployment Model chosen for the system?</td>
</tr>
<tr>
<td>Do all controls have at least one implementation status checkbox selected?</td>
</tr>
<tr>
<td>Are all critical controls implemented?</td>
</tr>
<tr>
<td>Are the customer responsibilities clearly identified in the CIS-CRM Tab, as well as the SSP Controls (by checkbox selected and in the implementation description)? Are the CIS-CRM and SSP Controls consistent for customer responsibilities? A sampling of several controls and associated controls identified in the text.</td>
</tr>
<tr>
<td>Does the Initial Authorization Agency concur with the CRM (adequacy and clarity of customer responsibilities)?</td>
</tr>
<tr>
<td>Does the Roles Table (User Roles and Privileges) sufficiently describe the range of user roles, responsibilities, and access privileges?</td>
</tr>
<tr>
<td>Is the control summary table, does the information in the responsible role correctly describe the required entities responsible for fulfilling the control? (With complete control mapping, we can track the work in SSP/Non-SSP Methods)</td>
</tr>
<tr>
<td>Is the appropriate Digital Identity Level selected?</td>
</tr>
<tr>
<td>Is the authorization boundary logically identified in the network diagram?</td>
</tr>
<tr>
<td>Does the CISO provide components to run on the client side?</td>
</tr>
<tr>
<td>Is there a data flow diagram that clearly illustrates the flow and protection of data going in and out of the service boundary and that includes at least four of both internal and external users?</td>
</tr>
<tr>
<td>Does the CISO use any third-party or external cloud services that lack FedRAMP Authorization? If yes, please list them.</td>
</tr>
<tr>
<td>Is the FedRAMP Authorization for this service? If Yes, are the &quot;overseas&quot; controls clearly identified in the control descriptions?</td>
</tr>
<tr>
<td>Are all interconnections correctly identified and documented in the SSP?</td>
</tr>
</tbody>
</table>
How does OSCAL Help?

Precise View of the Problem and Solution

“Complexity is irrelevant to the audience”
How does OSCAL Help?

**Consistency**

"Items are referred to by the same name or description throughout the document"

"Documentation issues"

"hot button issues"

"Not sure" "scary"
How does OSCAL Help?

Linear Timeline
Manual flows

vs

Parallel Processing
Auditable
Automated Flows
Reusable Flows/Rules
How does OSCAL Help?

**Budget:**
Staff time on prose and diagram review, consensus
Manual Risk and ConMon Vulnerability Analysis
Synchronous Work Streams

**vs**
Automation Pipelines and Algorithms
Async Work (Kanban)
How does OSCAL Help?

Increase Staff Efficiency and Engagement
Do more with automation
Requires New Skills and Culture*

Skill building vs paperwork:
Talent recruitment & retention

* Agency Leadership Champion
How does OSCAL Help?

Consistent & Verifiable Control Defn & Config

- Security Capability as Code ↔ Control Impls
- Control Parameters ↔ Profile Variables
- Control Implementation ↔ SAP Tests
- Traceability: Component ↔ Control ↔ Security Capability ↔ Threats
- Risk and Coverage Metrics
OSCAL Model-Based Assessment

Split your Brain for a Moment

- Models For Different Users
- Catalog ↔ Abstract Comp Defs
- SSP ↔ Control, Component Types
- Inventory ↔ Instances
- Coverage ↔ Risk Metrics exist at all levels
OSCAL Model-Based Risk Assessment

Package Review and Authorization Automation

- Validations **eliminate** SSP inconsistencies
- Component and Control code artifacts **deprecate** diagrams & narrative text
- Rules **connect** 3PAO Test Case Procedures to Controls => intent of a security control is met.
- TTPs **connect** SAR and POA&M findings to controls
1. Expect and Use OSCAL

PMO recommends CSP maturity...
Mature CSPs can use OSCAL now

Open source and vendor tools available (also for 3PAOs!*)

*CNCF, SunStone, others providing FOSS tools and support for any agency, 3PAO
HOWTO: OSCAL-Based SAF

2. Reskill and Retool

- Authorizing Official
  - OSCAL Package Assessor
  - OSCAL SSP/SAP/SAR Validation Automation
  - OSCAL ConMon Validation Automation
  - Policy Rules Administration
  - Policy Enforcement Scanner
- SME Rules Repository
  - OSCAL Control Repo
  - OSCAL Component Repo
  - Capability Test Repo
  - CWE, CVE, CTI, TTP Data
  - OSCAL Rules Repo
HOWTO: OSCAL-Based SAF

3. Use Algorithms Not Documentation

**Labels:** FIPS 199, TTPs/CVEs, Sec Comps, Data Flows

**Rules***: “What aspect of a part of system you evaluate. It may or may not relate to a control.”

**Tests***: “How do you specifically evaluate an aspect of a rule. It may or may not relate to a method for analyzing if control requirements are met.”

*Not the “Validation” Rules ie Schematron
4. Automate ALL Workflows

- Compliance-as-Code: Everything is OSCAL (or Tests)
- Compliance "Gitops" (aka "ComplianceOps") = PRs
- "Fork" Control Catalog, Agency provides Profiles
- CSP PRs component, capability, test defs
- 3PAOs PRs tests as rule code
- SAR, POA&M OSCAL PR checks
- ATOs as release gate in CI/CD
HOWTO: OSCAL-Based SAF

ASIDE: Parallel Workflows

- CSP CDef/Sec Cap/Test
- OSCAL Author
- Agency/CISA/Community Threat SMEs - Hardening and P/D/R Evaluations
- Agency Catalog+SSP Control Impl Rule Author
- Agency CISO/AO/MO PR Reviewers and Approvers
ASIDE: NISTIR 8011(ish)

**Desired State**

TTP ↔ D3FEND ↔ 800-53A ↔ SecCaps ↔ SAP

**“Declared” State**

(Policy ↔ (p)olicy ↔ Component Defs

**Monitor the actual state**

Asset Inventory ↔ SAR

Figure 4: HWAM Concept of Operations (CONOPS)
HOWTO: OSCAL-Based SAF

ASIDE: NISTIR 8011(ish) Prerequisites

Policy-as-Code (e.g. Rego)
Risk and Threat (e.g. TTP Heatmaps)
Enforcement Engine (e.g. OPA)
Test and Query Code Rules*

Future: ATO-as-Code

*https://github.com/usnistgov/OSCAL/issues/1058
ASIDE: Policy-as-Code

Desired state/behavior as code (aka policy-as-code)

- Example: OPA Rego
- Map TTPs -> Security Capabilities -> Control Requirements
- Generates Components Definitions (demo coming soon)
Threats and Vulnerabilities

ASIDE: Threat Guidance from PMO/.govCAR

- Protect, Detect, and Respond: MITRE ATT&CK TTPs
- % Implemented * Protection Value Score
- Prioritized Risk Profile
HOWTO: OSCAL-Based SAF

ASIDE:Configs-as-Code aka (p)olicy and SAP-as-Code

- Desired state/behavior as code ("declared" state)
- SAP queries actual system state via auto tests

TTPs ↔ Mitigations

- Security Capabilities ↔ Component
- Configs for each component ↔ Control Reqs
HOWTO: OSCAL-Based SAF

ASIDE: Assessment Engine

Open Policy Agent
- Uses DSL called Rego
  (you say ray-go I say ree-go)

Bored PhDs: SMT Solver (eg Microsoft Z3)
HOWTO: OSCAL-Based SAF

ASIDE: Threat and Queries

Threat-based analytic queries for “Defects”

- Narrative Keywords - NLP topics, semantic search
- Statistical Regression Analysis
- Graph Theory
- Heuristics
- AI/ML
5. Approve via “ComplianceOps”

- Compliance “Gitops” - PR all the things
- CSP: component, control updates as code
- 3PAOs: tests as query code
- Dry Run: Test Driven
- Drift Detection Policy and Alerts
- Coverage Metrics
- PR approvals gate ATO
Use Rules to Generate OSCAL

- **Desired State** ↔ Catalog: Controls, Profiles
- **Declared state** ↔ ComponentDefs, Security Capabilities
- **Threats** ↔ Control Implementation
- **Component Test Defs** ↔ SAP
- **Inventory** ↔ SAR
- **ProTip:** Operations on objects CRUD-y: use API tooling
Use Inventory+Threat Graph to Generate Code

Virtuous cycle: Controls, Tests Components, Mitigations

Future AI: eg. OpenAI Codex, RH/IBM Project Wisdom
HOWTO: OSCAL-Based SAF

FUTURE: ATO-as-Code

- Decision Gates (e.g. PR approvals and Actions) based on coded risk rubric (time, severity, impact)
- Time Machine Diffs
- MORE FUTURE: automated remediation PR generation
6. Continuous Updates

Catalog, Baseline, Profile Control Releases
ConMon continuous (SAP-defined) queries
CSP releases Component Defs
POA&M “Gitops” updates Tests
Think in Graphs (not lists)

- SSP is a control implementation ↔ component graph
- Relationships in the graph can encode threats, capabilities, tasks, responsibilities
- Graph queries encode model-based tests and coverage
- BONUS: Graph algorithms learn from new data AND structure
NLP: Map Control Objectives to Security Capabilities

Entity Extraction -> Semantic Search

Generate word/sentence embeddings using pre-trained language models eg. BERT/Hugging Face

Search for textually similar items - K-Nearest Neighbor (KNN)

Pros: Lots of text and very familiar

Cons: Not reusable; No test generation; Naming things is Hard!
OSCAL Algorithms

Generate Graph Queries

Pathfinding, Ontologies, Knowledge Graph
Test SSP Completeness, Coverage
Test SAP Completeness, Coverage
Test SAR, POA&M, Vulnerability Threat Exposure
OSCAL Algorithms

Learn vs Churn
New Threats, Vulnerabilities -> Graph Relationships and Tagging

MITRE ATT&CK -> Graph Query Predictions for Incident Response
Summary: ComplianceOps

OSCAL Compliance-as-Code

- Generate Component Defs and Tests
- Dry Run Test
- Drift Detection and Alerts
- Coverage Metrics
- PR approval gates
- Cattle* Not Pets

*No actual animals were harmed during the operation of compliance
Gaps and Roadblocks

Support for Automation

Vendors and FOSS Maintainers
APIs for ComponentDefs
APIs for Security Capabilities
Control Implementation Templates
APIs for TTPs, Mitigations, Scoring Algos/APIs
ML models and training data
Gaps and Roadblocks

**Community Support for Controls**

- More/better FOSS for Capability <-> Control Mapping
- More/better FOSS Tools for Control/Risk Assessment
- More/better FOSS OSCAL definitions

SAP schema improvements - Test Rules?
Gaps and Roadblocks

GRC Support for OSCAL
No CTRL+V from Word/Excel

3PAO Support for OSCAL
No more Ad Hoc Interviews and Screen Shots

CSP Support for OSCAL
Capability/Component Defs and Test Defs
Gaps and Roadblocks

Leadership and Mission Must Embrace Automation

Culture shift

Training and Skills Building

Systems (Graph) Thinking vs. Linear (List) Thinking
Questions? Free Stuff!

Github Repos Used in Demos

**SunStone Repo**
https://github.com/SunStone-Secure-LLC/oscal-for-agencies

**OPA Repo**
https://github.com/open-policy-agent/opa

**JupiterOne Starbase**
https://github.com/JupiterOne/starbase

@sunstonesecure-robert
robert@sunstonesecure.com

Need Help?
OSCAL Q&A Office Hours
threats contains analysis if {
    # preliminaries
    input.kind == "component-definition"
    
    # The 'some' keyword declares local variables.
    some capability in input.capabilities
    some prop in capability.props
    
    # define the guard rails for the control - what capabilities must it provide to be considered for a control impl?
    "T1699" == prop["ttp-id"]
    #prop["name"] == "ttt-id"
    #prop["value"] == "ttt-id"
    
    desc := sprint("Capability '%%' supports control AC-3 by addressing threat '%%': '%%',"
      [capability.name, prop["ttp-id"], prop["ttt-name"]],
    
    analysis := {"description": desc, "hmvalue": 12.6, "coverage": 0.05} # the coverage metric could come from actual meas
}

protect contains analysis if {
    # preliminaries
    input.kind == "component-definition"
    
    # The 'some' keyword declares local variables.
    some capability in input.capabilities
    some prop in capability.props
    
    # define the guard rails for the control - what capabilities must it provide to be considered for a control impl?
    "M1038" == prop["mitigation-id"]
    "implement-protect" == prop["govcar-category"]
    
    # score the implementation
    pw := pdrWeights[prop["govcar-category"]]
    pohe := ohes[prop["govcar-ohe"]]
    
    # every threat action that received a score for the associated control
control_implementations contains ci if {
  # preliminaries
  input.kind == "component-definition"
  protectionThreshold := 10.0

  # This magic can be augmented by generative code and solvers and ML.
  pscore := threats[.].hvvalue*protect[.].score+detect[.].score+respond[.].score
  pscore > protectionThreshold
  threats[.].coverage > .9

  # transform the cdf into impl
  some c in data.controls
  some s in c.statements
  stmt := { "statement-id": s["statement-id"], "uuid": s.uuid }

  cimpls := [
    { "uuid": c.uuid,
      "control-id": c["control-id"],
      "statements": [stmt],
      "rule-implinations": rule_implementations
    }
  ]

  # this comes from json in an OSCAL catalog
  ci := {
    "description": "This shows how to support control AC-3",
    "uid": "Feedbeef-0000-0000-0000-000000000000",
    "implemented-requirements": cimpls, #data.controls,
    "protection-value": pscore
  }
}
The Rego Playground

Rule implementations contain rule if {
  # preliminaries
  input.kind == "component-definition"
  # get the test specification attested to by the capability.
  some capability in input.capabilities
  some prop in t.props
  # define the guard rails for the control impl
  # what capabilities must be tested to be considered for a control impl?
  "inventory-class" == prop["assessment-protect-objective"]
  "Container" == prop["value"]
  # e.g. we only accept this control impl if it is tested for container signatures
  "component-capability" == t.props[0].name
  "signature" == t.props[0].value
  # note check the catalog to see if the test scenario condition is required
  some c in data.controls
  some s in c.statements
  some ts in s["testing-scenario-not-official-oscald-yet"]
  some cond in ts["conditions-for-sap-steps"]
  "required" == cond.props[0].name
  # this would be using more link structure but simple test for demo
  t.uuid == cond.uuid
  # generate the rules
  rule := {
    "description": "Test the signature value exists for a container image",
    # "props": t.props,
    # "prop": prop
    "condition": {
      "scenario": cond.title,
      "uuid": cond.uuid,
      "eval": "FIND Container WITH signatureValue == undefined"
    }
  }
  "capabilities": [
    { "name": "Prevent Unauthorized Containers",
      "props": []
    },
    "tests-not-official-oscald-yet": [
      { "uuid": "feedbeef-ac00-ac03-0001-000000000011",
        "props": [
          "assessment-protect-objective": "inventory-class",
          "value": "Container"
        ]
      }
    ]
  ]
}

"controls": [
  { "control-id": "ac-3",
    "statements": [
      { "statement-id": "ac-3_smt",
        "testing-scenario-not-official-oscald-yet": [
          { "conditions-for-sap-steps": [
            { "props": [
              { "name": "required",
                "value": "true"
              }
            ]
          },
          { "title": "Container with missing signature", "uuid": "feedbeef-ac00-ac03-0001-000000000011" }
        ]
      }
    ]
  }
]