National Institute of
Standards and Technology
U.S. Department of Commerce

Special Publication 800-126
Revision 1

The Technical Specification for the Security Content Automation Protocol (SCAP): SCAP Version 1.1

Recommendations of the National Institute of Standards and Technology

David Waltermire
Stephen Quinn
Karen Scarfone

NIST Special Publication 800-126
Revision 1

The Technical Specification for the Security Content Automation Protocol (SCAP): SCAP Version 1.1

Recommendations of the National Institute of Standards and Technology

David Waltermire
Stephen Quinn
Christopher Johnson
Karen Scarfone
John Banghart

C O M P U T E R S E C U R I T Y

Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-8930

February 2011

U.S. Department of Commerce

Gary Locke, Secretary

National Institute of Standards and Technology

Dr. Patrick D. Gallagher, Director

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analysis to advance the development and productive use of information technology. ITL's responsibilities include the development of technical, physical, administrative, and management standards and guidelines for the cost-effective security and privacy of sensitive unclassified information in Federal computer systems. This Special Publication 800-series reports on ITL's research, guidance, and outreach efforts in computer security and its collaborative activities with industry, government, and academic organizations.

National Institute of Standards and Technology Special Publication 800-126, Revision 1
39 pages (Feb. 2011)

Acknowledgments

The authors, David Waltermire and Stephen Quinn of the National Institute of Standards and Technology (NIST), and Karen Scarfone of G2, Inc., wish to thank their colleagues who reviewed drafts of this document and contributed to its technical content. The authors would like to acknowledge John Banghart, Harold Booth, and Paul Cichonski of NIST; Christopher Johnson of HP Enterprise Services; Paul Bartock of the National Security Agency (NSA); Jeff Ito, Matt Kerr, Shane Shaffer, and Greg Witte of G2, Inc.; Andy Bove of SecureAcuity; Jim Ronayne of Varen Technologies; Adam Halbardier, Rhonda Farrell, Angela Orebaugh, and Victoria Thompson of Booz Allen Hamilton; Alan Peltzman of the Defense Information Systems Agency (DISA); and Jon Baker, Drew Buttner, Maria Casipe, and Charles Schmidt of the MITRE Corporation for their keen and insightful assistance throughout the development of the document.

Trademark Information

Table of Contents

List of Tables and Figures

Executive Summary

The Security Content Automation Protocol (SCAP) is a suite of specifications that standardize the format and nomenclature by which security software products communicate software flaw and security configuration information[1]. SCAP is a multi-purpose protocol that supports automated configuration, vulnerability, and patch checking, technical control compliance activities, and security measurement. Goals for the development of SCAP include standardizing system security management, promoting interoperability of security products, and fostering the use of standard expressions of security content.

SCAP Version 1.1 is comprised of seven specifications—eXtensible Configuration Checklist Description Format (XCCDF), Open Vulnerability and Assessment Language (OVAL®), Open Checklist Interactive Language (OCIL), Common Platform Enumeration (CPE™), Common Configuration Enumeration (CCE™), Common Vulnerabilities and Exposures (CVE®), and Common Vulnerability Scoring System (CVSS). These specifications are grouped into three categories:

- **Languages.** The SCAP languages provide standard vocabularies and conventions for expressing security policy, technical check mechanisms, and assessment results.

- **Enumerations.** Each SCAP enumeration defines a standard nomenclature (naming format) and an official dictionary or list of items expressed using that nomenclature. For example, CVE provides a dictionary of publicly known information security vulnerabilities and exposures.[2]

- **Measurement and scoring systems.** In SCAP, this refers to evaluating specific characteristics of a vulnerability and, based on those characteristics, generating a score that reflects the vulnerability's severity.

SCAP utilizes software flaw and security configuration standard reference data, also known as *SCAP content*. This reference data is provided by the National Vulnerability Database (NVD),[3] which is managed by NIST and sponsored by the Department of Homeland Security (DHS).

This publication defines the technical composition of SCAP Version 1.1 in terms of its component specifications, their interrelationships, and the requirements for SCAP content, and also describes details of how the elements of SCAP interoperate. The technical specification describes the requirements and conventions that are to be employed to ensure the consistent and accurate exchange of SCAP content and the ability to reliably use the content with SCAP validated products.

The U.S. Federal Government, in cooperation with academia and private industry, is adopting SCAP and encourages its use in support of security automation activities and initiatives.[4] SCAP is achieving widespread adoption by major software and hardware manufacturers and has become a significant component of large information security management and governance programs. The protocol is expected to evolve and expand in support of the growing needs to define and measure effective security controls, assess and monitor ongoing aspects of that information security, and successfully manage systems in accordance with risk management frameworks such as NIST Special Publication 800-53[5], Department of Defense (DoD) Instruction 8500.2, and the Payment Card Industry (PCI) framework.

---

1 Products implementing SCAP can also be used to support non-security use cases such as configuration management and software inventory.

2 http://cve.mitre.org/

3 The National Vulnerability Database can be found at http://nvd.nist.gov/.

4 Refer to http://www.whitehouse.gov/omb/memoranda/fy2008/m08-22.pdf.

5 The Risk Management Framework is described in Section 3.0 of NIST Special Publication 800-53, available at http://csrc.nist.gov/publications/nistpubs/800-53-Rev3/sp800-53-rev3-final_updated-errata_05-01-2010.pdf

By detailing the specific and appropriate usage of the SCAP 1.1 components and their interoperability, NIST encourages the creation of reliable and pervasive SCAP content and the development of a wide array of products that leverage SCAP capabilities.

Organizations that develop SCAP 1.1-based content or products should implement the following recommendations:

Follow the requirements listed in this document and in the associated component specifications.

Organizations should ensure that their implementation and use of SCAP 1.1 is compliant with the requirements detailed in each component specification and the information presented in this document. If requirements are in conflict between component specifications, this document will provide clarification. If a component specification is in conflict with this document, the requirements in this document take precedence.

When creating SCAP content, adhere to the conventions specified in this document.

Security products and checklist authors assemble content from SCAP data repositories to create viable SCAP-expressed security guidance. A security configuration checklist that documents desired security configuration settings, installed patches, and other system security elements using SCAP in a standardized format is known as an SCAP-expressed checklist. Such a checklist would use XCCDF to describe the checklist, CCE to identify security configuration settings to be addressed or assessed, and CPE to identify platforms for which the checklist is valid. The use of CCE and CPE entries within XCCDF checklists is an example of an SCAP convention—a requirement for valid SCAP usage. These conventions are considered part of the definition of SCAP 1.1. Organizations producing SCAP content should adhere to these conventions to ensure the highest degree of interoperability.

1. Introduction

1.1 Authority

The National Institute of Standards and Technology (NIST) developed this document in furtherance of its statutory responsibilities under the Federal Information Security Management Act (FISMA) of 2002, Public Law 107-347.

NIST is responsible for developing standards and guidelines, including minimum requirements, for providing adequate information security for all agency operations and assets; but such standards and guidelines shall not apply to national security systems. This guideline is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130, Section 8b(3), "Securing Agency Information Systems," as analyzed in A-130, Appendix IV: Analysis of Key Sections. Supplemental information is provided in A-130, Appendix III.

This guideline has been prepared for use by Federal agencies. It may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright, though attribution is desired.

Nothing in this document should be taken to contradict standards and guidelines made mandatory and binding on Federal agencies by the Secretary of Commerce under statutory authority, nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other Federal official.

1.2 Purpose and Scope

This document provides the definitive technical specification for Version 1.1 of the Security Content Automation Protocol (SCAP). SCAP (pronounced ess-cap) consists of a suite of specifications for standardizing the format and nomenclature by which security software communicates information about software flaws and security configurations. This document defines SCAP requirements that are not defined in the individual SCAP component specifications. Each new requirement pertains either to using multiple component specifications together or to further constraining one of the individual component specifications. The requirements within the individual component specifications are not repeated in this document; see those specifications to access their requirements.

The scope of this document is limited to SCAP Version 1.1. Other versions of SCAP and its component specifications, including emerging specifications, are not addressed here. Future versions of SCAP will be defined in distinct revisions of this document, each clearly labeled with a document revision number and the appropriate SCAP version number. SCAP revisions are managed through a coordinated process defined within the SCAP Release Cycle.[6] The release cycle workflow manages changes related to SCAP specifications and validation processes including the addition of new specifications or updates to existing specifications. This process encourages community involvement, promotes transparency and awareness regarding proposed changes, and affords ample lead-time to prepare for pending changes.

1.3 Audience

This document is intended for three primary audiences:

- Content authors and editors seeking guidance to ensure that the SCAP content they produce operates correctly, consistently, and reliably in SCAP products.

[6] SCAP Release Cycle, http://scap.nist.gov/timeline.html

- Software developers and system integrators seeking to create, use, or exchange SCAP content in their products or service offerings.
- Product developers preparing for SCAP validation at an accredited independent testing laboratory.

This document assumes that readers already have general knowledge of SCAP and reasonable familiarity with the SCAP component specifications that their content, products, or services use. Individuals without this level of knowledge who would like to learn more about SCAP should consult NIST Special Publication (SP) 800-117, *Guide to Adopting and Using the Security Content Automation Protocol.*[7]

1.4 Document Structure

The remainder of this document is organized into the following major sections and appendices:

- Section 2 provides the high-level requirements for claiming conformance with the SCAP 1.1 specification.
- Section 3 details the requirements and recommendations for SCAP content syntax, structures, and development.
- Section 4 defines SCAP content processing requirements and recommendations.
- Section 5 provides additional requirements for selected common types of SCAP content.
- Appendix A contains an acronym and abbreviation list.
- Appendix B lists references and other resources related to SCAP 1.1.

1.5 Document Conventions

Some of the requirements and conventions used in this document reference eXtensible Markup Language (XML) content. These references come in two forms, inline and indented. An example of an inline reference is

"A `<cpe_dict:cpe-item>` may contain `<cpe_dict:check>` elements that reference OVAL Definitions".

In this example the notation `<cpe_dict:cpe-item>` can be replaced by the more verbose equivalent "the XML element whose qualified name is `cpe_dict:cpe-item`". An even more verbose equivalent is "the XML element in the namespace '*http://cpe.mitre.org/dictionary/2.0*' whose local name is `cpe-item`".

An example of an indented reference is:

> "References to OVAL Definitions are expressed using the following format:
>
> ```
> <cpe_dict:check system=
> "http://oval.mitre.org/XMLSchema/oval-definitions-5"
> href="Oval_URL">[Oval_inventory_definition_id]
> </cpe_dict:check>".
> ```

[7] http://csrc.nist.gov/publications/PubsSPs.html

The general convention used when describing XML attributes within this document is to reference the attribute as well as its associated element including the namespace alias, employing the general form: "`@attributeName` for the `<prefix:localName>`".

Indented references are intended to represent the form of actual XML content. Indented references represent literal content by the use of a `fixed-length font`, and parametric (freely replaceable) content by the use of an *italic font*. Square brackets '`[]`' are used to designate optional content. Thus "`[`*Oval_inventory_definition_id*`]`" designates optional parametric content.

Both inline and indented forms use qualified names to refer to specific XML elements. A qualified name associates a named element with a namespace. The namespace identifies the specific XML schema that defines (and consequently may be used to validate) the syntax of the element instance. A qualified name declares this schema to element association using the format '*prefix*:*element-name*'. The association of prefix to namespace is defined in the metadata of an XML document and generally will vary from document to document. In this specification, the conventional mappings listed in Table 1 are used.

Table 1. Conventional XML Mappings

Prefix	Namespace URI	Schema
cpe	http://cpe.mitre.org/language/2.0	Embedded CPE references
cpe_dict	http://cpe.mitre.org/dictionary/2.0	CPE Dictionaries
cve	http://scap.nist.gov/schema/vulnerability/0.4	NVD/CVE data feed elements and attributes
cvss	http://scap.nist.gov/schema/cvss-v2/0.2	NVD/CVSS data feed elements and attributes
dc	http://purl.org/dc/elements/1.1/	Simple Dublin Core elements
ds	http://www.w3.org/2000/09/xmldsig#	Interoperable XML digital signatures
nvd	http://scap.nist.gov/schema/feed/vulnerability/2.0	Base schema for NVD data feeds
ocil	http://scap.nist.gov/schema/ocil/2.0	OCIL elements and attributes
oval	http://oval.mitre.org/XMLSchema/oval-common-5	Common OVAL elements and attributes
oval-def	http://oval.mitre.org/XMLSchema/oval-definitions-5	OVAL Definitions
oval-res	http://oval.mitre.org/XMLSchema/oval-results-5	OVAL results
oval-sc	http://oval.mitre.org/XMLSchema/oval-system-characteristics-5	OVAL system characteristics
oval-var	http://oval.mitre.org/XMLSchema/oval-variables-5	The elements, types, and attributes that compose the core schema for encoding OVAL Variables. This schema is provided to give structure to any external variables and their values that an OVAL Definition is expecting.
sch	http://purl.oclc.org/dsdl/schematron	Schematron validation scripts
xccdf	http://checklists.nist.gov/xccdf/1.1	XCCDF policy documents
xml	http://www.w3.org/XML/1998/namespace	Common XML attributes
xxxx-def	http://oval.mitre.org/XMLSchema/oval-definitions-5#*xxxx*	OVAL elements and attributes specific to an OS, Hardware, or Application type *xxxx*[8]
xxxx-sc	http://oval.mitre.org/XMLSchema/oval-system-characteristics-5#*xxxx*	OVAL system characteristic elements and attributes specific to an OS, Hardware, or Application type *xxxx*

[8] The types supported by OVAL 5.3 include the AIX, CATOS, ESX, FREE BSD, HP-UX, IOS, LINUX, PIXOS, SOLARIS, UNIX, WINDOWS, INDEPENDENT (common) operating systems, and APACHE application.

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in Request for Comment (RFC) 2119.[9]

[9] RFC 2119, “Key words for use in RFCs to Indicate Requirement Levels”, is available at http://www.ietf.org/rfc/rfc2119.txt.

2. SCAP 1.1 Conformance

SCAP 1.1 uses the following specifications:

- Extensible Configuration Checklist Description Format (XCCDF) 1.1.4, a language for authoring security checklists/benchmarks and for reporting results of checklist evaluation [XCCDF]
- Open Vulnerability and Assessment Language (OVAL) 5.8, a language for representing system configuration information, assessing machine state, and reporting assessment results [OVAL]
- Open Checklist Interactive Language (OCIL) 2.0, a language for representing checks that collect information from people or from existing data stores made by other data collection efforts [OCIL]
- Common Platform Enumeration (CPE) 2.2, a nomenclature and dictionary of hardware, operating systems, and applications [CPE]
- Common Configuration Enumeration (CCE) 5, a nomenclature and dictionary of software security configurations [CCE]
- Common Vulnerabilities and Exposures (CVE), a nomenclature and dictionary of security-related software flaws[10] [CVE]
- Common Vulnerability Scoring System (CVSS) 2.0, a specification for measuring the relative severity of software flaw vulnerabilities [CVSS].

All references to these specifications within this document are to the version numbers listed above, unless otherwise explicitly specified.

Combinations of these specifications can be used together for particular functions, such as security configuration scanning. These functions, known as *SCAP capabilities*, are not product types, but rather ways in which a product can use SCAP. The collective XML content used for a capability is called an *SCAP data stream*. An *SCAP source data stream* holds the input content, and an *SCAP result data stream* holds the output content. The major elements of a data stream, such as the XCCDF portion or the OVAL patch portion, are referred to as *stream components*.

Products and organizations may want to claim conformance to one or more of the SCAP capabilities within the SCAP 1.1 specification for a variety of reasons. For example, a product may want to assert that it uses SCAP content properly and can interoperate with other products using proper SCAP content. Another example is a policy mandating that an organization use SCAP for performing vulnerability assessments and other security operations.

This section provides the high-level requirements that a product or SCAP content must meet for conformance with the SCAP 1.1 specification. Most of the requirements listed in this section reference other sections in the document that fully define the requirements.

2.1 Product Conformance

All IT products claiming conformance with the SCAP 1.1 specification SHALL adhere to the following requirements:

1. Adhere to the requirements detailed in each applicable component specification (for each selected

[10] CVE does not have a version number.

SCAP component specification, and for each SCAP component specification required to implement the selected SCAP capabilities). The authoritative references for each specification are listed in Appendix B. If requirements are in conflict between component specifications, this document will provide clarification. If a component specification is in conflict with this document, the requirements in this document SHALL take precedence.

2. Products that process SCAP data streams SHALL consume and correctly process well-formed SCAP data streams. This includes following all of the processing requirements defined in Section 4 for each selected SCAP component specification and for each SCAP component specification required to implement the selected SCAP capabilities.

3. Products that produce SCAP data streams SHALL produce well-formed SCAP data streams. This includes following all of the syntax, structural, and other content design requirements defined in Section 3 for each selected SCAP component specification and for each SCAP component specification required to implement the selected SCAP capabilities. This also includes following the requirements in Section 5 if the content is one of the types addressed in that section.

4. Make an explicit claim of conformance to this specification in any documentation provided to end users.

2.2 Organization Conformance

Organizations creating or maintaining SCAP data streams that claim conformance with the SCAP 1.1 specification SHALL adhere to the following requirements:

1. Adhere to the requirements detailed in each applicable component specification (for each selected SCAP component specification, and for each SCAP component specification required to implement the selected SCAP capabilities). The authoritative references for each specification are listed in Appendix B. If requirements are in conflict between component specifications, this document will provide clarification. If a component specification is in conflict with this document, the requirements in this document SHALL take precedence.

2. Follow all of the syntax, structural, and other content design requirements defined in Section 3 for each selected SCAP component specification and for each SCAP component specification required to implement the selected SCAP capabilities. If the content is one of the types addressed in Section 5, follow all of its requirements as well.

3. SCAP Content Requirements and Recommendations

This section defines the content syntax, structure, and development requirements that products and content authors and editors MUST follow to produce valid SCAP 1.1 content. This section also provides recommendations that are not mandatory; organizations are encouraged to adopt them to promote stronger interoperability and greater consistency. The first part of the section discusses SCAP source data stream requirements. The rest of the section groups requirements and recommendations by specification: XCCDF, OVAL, OCIL, CPE, CCE, CVE, and CVSS, in that order.

3.1 SCAP Source Data Streams

An SCAP data stream is a collection of XML instance documents, also called stream components. The required XML content composing an SCAP data stream depends on the use case and is designed to satisfy specific policy or situational awareness objectives. There are two types of SCAP data streams: source and result. Section 4.5 discusses SCAP result data streams, which contain the results that are generated during processing.

An SCAP source data stream is the expression of content for a specific use case using one or more stream components. For its filenames, every SCAP source data stream SHALL use a common locator prefix that is appended to the URL base of the deployed data source. Every SCAP source data stream component SHALL have a filename comprised of the locator prefix (including a trailing hyphen) followed by the appropriate component suffix, as listed in Table 2.

Table 2. SCAP Source Data Stream Conventions

Component	Component Suffix	Document Element
XCCDF Benchmark	xccdf.xml	`<xccdf:Benchmark>`
OVAL Compliance	oval.xml	`<oval-def:oval_definitions>`
OVAL Patch	patches.xml	`<oval-def:oval_definitions>`
OVAL Vulnerability	oval.xml	`<oval-def:oval_definitions>`
OCIL Questionnaire	ocil.xml	`<ocil:ocil>`
CPE Dictionary	cpe-dictionary.xml	`<cpe-dict:cpe-list>`
CPE Inventory	cpe-oval.xml	`<oval-def:oval_definitions>`

For example:

```
file:///c:/content/example-winxp-xccdf.xml

The URL base is: file:///c:/content/
The locator prefix is: example-winxp-
The stream component is: xccdf.xml
```

Table 2 also lists document elements. Each SCAP source data stream component SHALL use the specified element as its document element.

Each SCAP source data stream component SHALL validate against the corresponding schema and, if applicable, associated Schematron style sheet. Each SCAP source data stream component SHOULD NOT use any constructs that are deprecated in its associated specification. Validation of each component SHALL be done in accordance with the portions of this document that define requirements for the

component. NIST provides an SCAP Content Validation Tool, which is designed to help validate the correctness of an SCAP source data stream.[11] The SCAP Content Validation Tool is a simple command-line tool that will check that SCAP content is well-formed, all cross references are valid, and required values are appropriately set. All errors and warnings are returned in both XML and Hypertext Markup Language (HTML) formats.

3.2 XCCDF

This section lists the eXtensible Configuration Checklist Description Format (XCCDF) requirements and recommendations. They are organized by the following categories: general, *<xccdf:platform>*, *<xccdf:Benchmark>*, *<xccdf:Profile>*, *<xccdf:Rule>*, and check system usage.

3.2.1 General

The following general restrictions apply to SCAP XCCDF content:

1. The use of the *@xml:base* attribute SHALL NOT be allowed. This attribute is not compatible with the SCAP data stream model.

2. The *<xccdf:Benchmark>* element SHALL have an *@xml:lang* attribute.

3. If an *@xml:lang* attribute is omitted within the content model, the *@xml:lang* attribute of the nearest ancestor element that has the attribute defined SHALL be consulted. Possible ancestor elements are *<xccdf:Value>*, *<xccdf:Group>*, *<xccdf:Rule>*, and *<xccdf:Benchmark>*.

XCCDF metadata is used by SCAP products to assist in the selection of the appropriate SCAP data stream, ensure that the most recent or correct version of an XCCDF document is used, and provide additional information about the document. The following metadata requirements and conventions apply to the *<xccdf:Benchmark>*, *<xccdf:Profile>*, *<xccdf:Value>*, *<xccdf:Group>*, and *<xccdf:Rule>* elements:

1. One or more instances of the *<xccdf:title>* element SHALL be provided. Each instance MUST contain a text value that indicates the purpose of the containing element and MAY include the OPTIONAL *@xml:lang* attribute. If more than one *<xccdf:title>* element is provided, the *@xml:lang* attribute SHALL be provided.

2. One or more instances of the *<xccdf:description>* element SHALL be provided. Each instance MUST contain text values that represent the purpose of the containing element and MAY include the OPTIONAL *@xml:lang* attribute. If more than one *<xccdf:description>* element is provided, the *@xml:lang* attribute SHALL be provided.

All remaining OPTIONAL elements in the XCCDF schema MAY be included at the author's discretion unless otherwise noted in this document.

[11] The tool can be downloaded from http://scap.nist.gov/revision/1.1/index.html#tools.

3.2.2 The <xccdf:platform> Element and CPE Names

For all SCAP content, the applicability of *<xccdf:Benchmark>*, *<xccdf:Profile>*, *<xccdf:Group>*, and *<xccdf:Rule>* elements to specific IT platforms MAY be specified using one or more *<xccdf:platform>* *@idref* attributes. Each instance of the *@idref* attribute SHALL reference either a CPE Name or the `@id` attribute of a *<cpe-lang:platform-specification/cpe-lang:platform>* element.

If compound CPE Name statements are necessary, a CPE Language *<cpe-lang:platform-specification>* element SHALL be defined as a child of the *<xccdf:Benchmark>* element. The *@id* attribute for each *<cpe-lang:platform>* element declared in this manner MAY be referenced within an *<xccdf:platform>* element with a corresponding *@idref* attribute. Complex platforms MAY be referenced this way within *<xccdf:Benchmark>*, *<xccdf:Profile>*, *<xccdf:Group>*, and *<xccdf:Rule>* elements.

For example:

```
<cpe-lang:platform-specification>
  <cpe-lang:platform id="xp_and_acrobat">
    <cpe-lang:logical-test operator="AND" negate="false">
      <cpe-lang:fact-ref name="cpe:/o:microsoft:windows_xp"/>
      <cpe-lang:fact-ref name="cpe:/a:adobe:acrobat:7.0.9"/>
    </cpe-lang:logical-test>
  </cpe-lang:platform>
</cpe-lang:platform-specification>
<xccdf:platform idref="xp_and_acrobat"/>
```

Within a given *<xccdf:Benchmark>*, *<xccdf:Profile>*, *<xccdf:Group>*, or *<xccdf:Rule>* context, if no *<xccdf:platform>* element is defined, the *<xccdf:platform>* of its nearest ancestor that has an *<xccdf:platform>* element defined SHALL be inherited. If none of its ancestors have an *<xccdf:platform>* element defined, the *<xccdf:Benchmark>*, *<xccdf:Profile>*, *<xccdf:Group>*, or *<xccdf:Rule>* SHALL be considered to apply to any product.

CPE Names used within an XCCDF document SHALL match the names of existing Official CPE Dictionary[12] entries where names for the desired platform exist. The matching algorithm from [CPE] to be used SHALL be CPE_Name_Match for a single CPE Name and CPE_Language_Match for a compound CPE Name. If multiple matches are found within the dictionary (e.g., deprecated and current CPE Names), the most current CPE Name SHOULD be used.

Each reference to a CPE Name SHALL be declared in the required CPE dictionary data stream component, and each OVAL inventory class definition referenced from the dictionary data stream component SHALL be specified in the required CPE inventory data stream component.

3.2.3 The <xccdf:Benchmark> Element

The following requirements and recommendations apply to the *<xccdf:Benchmark>* element:

[12] The Official CPE Dictionary is located at http://nvd.nist.gov/cpe.cfm.

1. The REQUIRED *@id* attribute SHALL be used to uniquely identify all revisions of a benchmark. Multiple revisions of a single benchmark SHOULD have identical identifiers, so that someone who reviews the revisions can readily identify them as multiple versions of a single benchmark.

2. The *@style* attribute SHALL have the value "SCAP_1.1".

3. The *<xccdf:status>* element SHALL indicate the current status of the benchmark document. The associated text value SHALL be "draft" for documents released in public draft state and "accepted" for documents that have been officially released by an organization. The *@date* attribute SHALL be populated with the date of the status change. Additional *<xccdf:status>* elements MAY be included to indicate historic status transitions.

4. The *<xccdf:version>* element SHALL uniquely identify the particular revision of the benchmark. Also, these revisions SHOULD have version values that indicate the revision sequence, so that the history of changes from the original benchmark can be determined. The *@time* attribute of the *<xccdf:version>* element SHOULD be used for a timestamp of when the benchmark was defined. The *@update* attribute of the *<xccdf:version>* element SHOULD be used for a URI that specifies where updates to the benchmark can be obtained.

5. One or more instances of the *<xccdf:notice>* element MAY be provided indicating clarifications, suggestions, or warnings regarding the use of the benchmark, including but not limited to terms of use, legal notices, or copyright statements.

6. The *<xccdf:metadata>* element SHALL be provided and SHALL, at minimum, contain the Dublin Core[13] terms from Table 3. Additional Dublin Core terms SHALL follow the required terms within the element sequence.

Table 3. Use of Dublin Core Terms in XCCDF Metadata

Dublin Core Term	Description of Use
<dc:creator>	The person, organization, and/or service that created the XCCDF XML instance
<dc:publisher>	The person, organization, and/or service that published the XCCDF XML instance
<dc:contributor>	The person, organization, and/or service that contributed to the creation of the XCCDF XML instance
<dc:source>	An identifier that indicates the organizational context of the *<xccdf:Benchmark>* element's *@id* attribute. An organizationally specific URI SHOULD be used.

3.2.4 The <xccdf:Profile> Element

The use of an *<xccdf:Profile>* element SHALL NOT be required. SCAP content commonly includes *<xccdf:Profile>* elements, but they are optional.

3.2.5 Allowed Check System Usage

The following requirements and recommendations apply to the use of the *<xccdf:check>* and *<xccdf:complex-check>* elements:

[13] http://dublincore.org/documents/dces/

1. The *<xccdf:check-content>* element SHALL NOT be used to embed check content directly into XCCDF content.

2. At least one *<xccdf:check-content-ref>* element MUST be provided for each *<xccdf:check>*.

3. Use of XCCDF check systems as specified in the *<xccdf:check>* element's *@system* attribute SHALL be restricted as follows:

   a. The following check systems are *supported* by SCAP:

      i. Use of the OVAL check system SHALL be indicated by the *http://oval.mitre.org/XMLSchema/oval-definitions-5* system identifier.

      ii. Use of the OCIL check system SHALL be indicated by the *http://scap.nist.gov/schema/ocil/2* system identifier.

   b. If a check system is used in XCCDF content that is not *supported* by SCAP, then this content SHALL NOT be considered *well-formed* with regards to SCAP.

If multiple *<xccdf:check-content-ref>* elements occur within an *<xccdf:check>* element, the *<xccdf:check-content-ref>* elements are evaluated in the order they appear. The first resolvable *<xccdf:check-content-ref>* element is used to determine the *<xccdf:Rule>* status. For each *<xccdf:check-content-ref>* element, an implementation attempts to retrieve the document referenced by the element's `@href` attribute. If not resolvable, the next available *<xccdf:check-content-ref>* element is evaluated. If none of the *<xccdf:check-content-ref>* elements are resolvable, then the result of the rule evaluation is the XCCDF "unchecked" status and processing of the *<xccdf:Rule>* ends. The `@href` attribute MAY map a remote URL to a local copy of the file in cases where remote access is not available, allowed, or practical.

3.2.5.1 OVAL <xccdf:check> Usage

References from SCAP compliant XCCDF to OVAL Definitions SHALL use the form:

```
<check-content-ref href="OVAL_Source_URI" [name="OVAL_Definition_Id"]/>
```

The *@href* attribute SHALL reference an OVAL source data stream component. When present, the *@name* attribute SHALL refer to a specific OVAL Definition in the designated source data stream component. Use of the *@name* attribute is REQUIRED except for the patches up-to-date rule, as defined in Section 3.2.6.4.

In the previous example, the *<xccdf:check-content-ref>* element's *@href* attribute refers to an OVAL Definition source data stream component containing one or more OVAL patch definitions. This *<xccdf:check-content-ref>* is equivalent to *referencing* a virtual OVAL Definition of the form:

```
<oval_definitions xmlns:oval-def="http://oval.mitre.org/XMLSchema/oval-definitions-5">
   <definitions>
      <definition id="identifier of patch definition" version="0" class="patch">
         …
```

```
            <criteria>
               <extend_definition definition_ref="identifier of patch definition 1"/>
               …
               <extend_definition definition_ref="identifier of patch definition N"/>
            </criteria>
         </definition>
      </definitions>
   </oval_definitions>
```

where the extended definitions are the individual patch definitions defined in the OVAL source data stream component.

See Section 4.6.3 for additional information on mapping OVAL results to XCCDF results.

3.2.5.2 <xccdf:Value> and OVAL Variable Dependencies

One or more *<xccdf:check-export>* elements MAY be used to define the binding of *<xccdf:Value>* elements to OVAL variables. The format of the *<xccdf:check-export>* element is:

```
<xccdf:check-export value-id="XCCDF_Value_id"
export-name="OVAL_External_Variable_id"/>
```

The following check element example demonstrates the use of this convention:

```
<xccdf:check system="http://oval.mitre.org/XMLSchema/oval-definitions-5">
   <xccdf:check-export value-id="NoSlowLink_var"
   export-name="oval:gov.nist.fdcc.xp:var:66711"/>
   <xccdf:check-export value-id="NoBackgroundPolicy_var"
   export-name="oval:gov.nist.fdcc.xp:var:66712"/>
   <xccdf:check-export value-id="NoGPOListChanges_var"
   export-name="oval:gov.nist.fdcc.xp:var:66713"/>
   <xccdf:check-content-ref href="fdcc-winxp-oval.xml"
   name="oval:gov.nist.fdcc.xp:def:6671"/>
</xccdf:check>
```

The type and value binding of the specified XCCDF Value is constrained to match that lexical representation of the indicated OVAL Variable Data Type. Table 4 summarizes the constraints regarding data type usage. Additional information regarding OVAL and XCCDF data types can be found in the OVAL Common Schema documentation[14] and the XCCDF specification [XCCDF].

Table 4. XCCDF-OVAL Data Export Matching Constraints

OVAL Data Type	Matching XCCDF Data Type
int	number
float	number
boolean	boolean

[14] http://oval.mitre.org/language/download/schema/version5.4/ovaldefinition/documentation/oval-common-schema.html#DatatypeEnumeration and http://oval.mitre.org/language/download/schema/version5.3/ovaldefinition/documentation/oval-definitions-schema.pdf

OVAL Data Type	Matching XCCDF Data Type
string, evr_string, version, ios_version, fileset_revision, binary	string

3.2.5.3 OCIL <xccdf:check> Usage

When referencing OCIL questionnaires as checks, XCCDF content SHALL follow all requirements defined in Appendix B of NIST Interagency Report (IR) 7692, *Specifications for the Open Checklist Interactive Language (OCIL) Version 2.0* [OCIL].

3.2.6 The <xccdf:Rule> Element

The following requirements and recommendations apply to the *<xccdf:Rule>* element.

3.2.6.1 The <xccdf:ident> Element

Each *<xccdf:Rule>* element SHALL include an *<xccdf:ident>* element containing a CVE, CCE, or CPE identifier reference if an appropriate reference exists. If the rule references an OVAL definition, then *<xccdf:ident>* element content SHALL match the corresponding CVE, CCE, or CPE identifier found in the associated OVAL Definition(s) if an appropriate identifier exists.

When referencing a CVE, CCE, or CPE identifier:

1. The identifier type SHALL correspond to the OVAL definition class, as follows:
   a. OVAL compliance class definitions reference CCE identifiers.
   b. OVAL inventory class definitions reference CPE identifiers.
   c. OVAL patch and vulnerability class definitions reference CVE identifiers.
2. The system attribute for the *<xccdf:ident>* element SHALL be defined using one of the following:
   a. The CVE system identifier, either "CVE" or "*http://cve.mitre.org"* (preferred method)
   b. The CCE system identifier, either "CCE" or "*http://cce.mitre.org"* (preferred method)
   c. The CPE system identifier, either "CPE" or *"http://cpe.mitre.org"* (preferred method)

For example:

```
<Rule id="AuditAccountLogonEvents">
    …
    <ident system="http://cce.mitre.org">CCE-3867-0</ident>
    …
</Rule>
```

An *<xccdf:ident>* element referencing a CVE, CCE, or CPE identifier SHALL be ordered before other *<xccdf:ident>* elements referencing non-SCAP identifiers. Identifiers from previous revisions of CCE or CPE MAY also be specified following the SCAP identifiers.

3.2.6.2 OVAL Definition References

If an *<xccdf:Rule>* element references a specific OVAL Definition, then:

1. The referenced OVAL Definition MUST have its *@class* attribute defined as "compliance" if it represents a check for the value of a specific configuration setting.
2. The referenced OVAL Definition MUST have its *@class* attribute defined as "vulnerability" if it represents a check for the presence of a particular software flaw vulnerability.
3. The referenced OVAL Definition MUST have its *@class* attribute defined as "patch" if it represents a check for the presence of a discrete patch.
4. The referenced OVAL Definition MUST have its *@class* attribute defined as "inventory" if it represents a check for the presence of a product of interest.

3.2.6.3 OCIL Questionnaire References

An XCCDF rule MAY reference an OCIL questionnaire. This SHOULD be done only for cases where OVAL cannot perform the check.

3.2.6.4 Use of a Patches Up-To-Date Rule

An OVAL instance document MAY be used to represent a series of checks to verify that patches have been installed. Historically, an XCCDF convention has been used to identify such a reference. An XCCDF benchmark MAY include a patches up-to-date rule that references an OVAL patch source data stream component. When implementing a patches up-to-date XCCDF rule, the following approach SHALL be used:

1. The source data stream MUST include an OVAL Patch source data stream component.
2. The *<xccdf:Rule>* element that references an OVAL Patch source data stream component SHALL have the *@id* attribute value of "*security_patches_up_to_date*".
3. A single *<xccdf:check>* element SHALL be provided for the *<xccdf:Rule>* with a *@system* attribute value of "*http://oval.mitre.org/XMLSchema/oval-definitions-5"*.
4. Each *<xccdf:check-content-ref>* element SHALL have an *@href* attribute referencing a valid SCAP *<oval-def:oval_definitions>* document instance with the *@name* attribute omitted.

For example:

```
<Rule id="security_patches_up_to_date" selected="false">
   <title>Security Patches Up-To-Date</title>
   <description>Keep systems up to current patch levels</description>
   <check system="http://oval.mitre.org/XMLSchema/oval-definitions-5">
      <check-content-ref href="scap-win2000-patches.xml"/>
   </check>
</Rule>
```

3.2.6.5 CVSS Scores

SCAP 1.0 required the inclusion of static CVSS scores in XCCDF vulnerability-related rules. However, CVSS base scores sometimes change over time, such as when more information is available about a particular vulnerability, and CVSS temporal and environmental scores are intended to change to reflect current threats, security controls, and other factors. Current CVSS scores acquired dynamically, such as from a data feed, SHOULD be used in place of static CVSS scores in the `@weight` attribute within XCCDF vulnerability-related rules. Section 3.8 contains additional requirements for CVSS usage.

3.3 OVAL

While the default version[15] of OVAL used in SCAP 1.1 SHALL be OVAL version 5.8, content authors SHOULD utilize the earliest SCAP-supported version of OVAL (5.3 at minimum) that includes all required tests and is necessary to properly address the content's purpose or use case. This approach, often referred to as the "least-version-principle", allows for SCAP content to remain viable over a longer period of time by allowing for the broadest support within products, while reducing the content maintenance burden that would be required to maintain revisions of content for multiple specification versions.

All of the OVAL content MUST contain an *<oval:generator>* element. The version of any particular document instance SHALL be specified using the *<oval:schema_version>* content element of the *<oval:generator>* as in this example:

```
<oval:generator>
  <oval:product_name>The OVAL Repository</oval:product_name>
  <oval:schema_version>5.8</oval:schema_version>
</oval:generator>
```

The version of an *<oval-var:oval_variables>* document SHALL be the same as that of the *<oval-def:oval_definitions>* document whose external variables are bound by the variables document.

The following requirements apply to particular classes of OVAL definitions:

1. For compliance class definitions:
    a. If an OVAL compliance class definition maps to one or more CCE identifiers, the definition SHOULD include *<oval-def:reference>* elements that reference those identifiers using the following format:

        ```
        <oval-def:reference source="http://cce.mitre.org"
        ref_id="CCE_identifier"/>
        ```

        The source attribute SHALL be defined using either "CCE" or "*http://cce.mitre.org"* (preferred method).

    b. Definitions that are directly or indirectly extended SHALL be limited to inventory and compliance classes.
2. For inventory class definitions:

[15] The OVAL Language versioning methodology is available here: http://oval.mitre.org/language/about/versioning.html

a. If an OVAL inventory class definition maps to one or more CPE identifiers, the definition SHOULD include *<oval-def:reference>* elements that reference those identifiers using the following format:

```
<oval-def:reference source="http://cpe.mitre.org"
ref_id="CPE_identifier"/>
```

The source attribute SHALL be defined using either "CPE" or "*http://cpe.mitre.org"* (preferred method).

b. Definitions that are directly or indirectly extended SHALL be limited to the inventory class.

3. For patch class definitions:

a. If an OVAL patch class definition maps to one or more CVE identifiers, the definition MAY include *<oval-def:reference>* elements that reference those identifiers using the following format:

```
<oval-def:reference source="http://cve.mitre.org"
ref_id="CVE_identifier"/>
```

The source attribute SHALL be defined using either "CVE" or "*http://cve.mitre.org"* (preferred method).

b. If an OVAL patch class definition is associated with a source specific identifier (for example, Knowledge Base numbers for Microsoft patches), these identifiers SHOULD be included in *<oval-def:reference>* elements contained by the definition. For example:

```
<oval-def:reference source="www.microsoft.com/Patch"
ref_id="KB912919"/>
```

c. Definitions that are directly or indirectly extended SHALL be limited to inventory and patch classes.

4. For vulnerability class definitions:

a. If an OVAL vulnerability class definition maps to one or more CVE identifiers, the definition SHOULD include *<oval-def:reference>* elements that reference those identifiers using the following format:

```
<oval-def:reference source="http://cve.mitre.org"
ref_id="CVE_identifier"/>
```

The source attribute SHALL be defined using either "CVE" or "*http://cve.mitre.org"* (preferred method).

b. Definitions that are directly or indirectly extended SHALL be limited to inventory and vulnerability classes.

3.4 OCIL

OCIL content SHOULD be used for checking rules that cannot be fully automated with OVAL. For example, a particular software product may not have an application programming interface (API) that supports OVAL use. Another example is performing a check that requires user interaction, such as asking

the user to look up information within a management console or to report a serial number affixed to a computing device. OCIL can also be used to collect a user's information, such as whether the user participated in a recent security training session.

3.5 CPE

The Official CPE Dictionary data feed[16] MAY be used by SCAP components to reference CPE Names. Local enumerations are permitted, but if a CPE Name for a product or platform exists in the Official CPE Dictionary, the content SHALL match the product or platform referenced by that official identifier.

Section 8 of [CPE] provides the defining structure of the Official CPE Dictionary. For certain names, a *<cpe_dict:cpe-item>* MAY contain one or more *<check>* elements that reference OVAL system inventory definitions using the following format:

```
<cpe_dict:check system="http://oval.mitre.org/XMLSchema/oval-definitions-5"
      [href="oval_URL"]>oval_inventory_definition_id</cpe_dict:check>
```

For example:

```
<cpe-list xmlns="http://cpe.mitre.org/dictionary/2.0"
          xmlns:cpe_dict="http://cpe.mitre.org/dictionary/2.0">
   <cpe-item name="cpe:/o:microsoft:windows_2003">
      <title>Microsoft Windows Server 2003</title>
      <check system=http://oval.mitre.org/XMLSchema/oval-definitions-5
             href="example-winsvr2003-oval.xml">
             oval:org.mitre.oval:def:128
      </check>
   </cpe-item>
</cpe-list>
```

The referenced OVAL inventory definition SHALL specify the technical procedure for determining whether or not a specific target asset is an instance of the CPE Name specified by the *<cpe_dict:cpe-item>* element. This usage is encouraged for a CPE dictionary source data stream component.

If a *<cpe_dict:cpe-item>* contained in a CPE dictionary data stream component references an OVAL "inventory" definition, then that definition SHALL be resolved by an *@href* attribute referencing a CPE Inventory source data stream component in the same data stream. Furthermore, the title of the *<cpe_dict:cpe-item>* SHALL match the title of an affected platform bound to the referenced definition.

3.6 CCE

To maintain consistency and accuracy among SCAP content, SCAP content referencing a configuration setting SHALL use the official CCE identifier if a CCE entry for a particular configuration setting exists in the Official CCE Dictionary. If no CCE exists for the configuration setting of interest, the content author SHOULD seek to have a CCE identifier issued for the configuration setting. See the OVAL compliance class definition requirements in Section 3.3 and the `<xccdf:ident>` requirements in Section 3.2.6.1 for additional requirements involving CCE identifier references.

[16] The Official CPE Dictionary is located at http://nvd.nist.gov/cpe.cfm.

The MITRE Corporation maintains the current official CCE list at http://cce.mitre.org/lists/cce_list.html and new CCEs can be requested from the CCE Content Team at http://cce.mitre.org/lists/creation_process.html.

Use of an official, dynamic data feed is preferred to static coding of values in SCAP data sources. The NVD provides a data feed[17] that correlates CCE identifiers with the control identifiers described in NIST SP 800-53. Embedding control identifiers within SCAP content is strongly discouraged due to the maintenance burden that it imposes on content maintainers when the control identifiers are revised.

3.7 CVE

CVE references in SCAP content MAY include both "candidate" and "entry" status identifiers. The use of deprecated CVE identifiers SHALL NOT be allowed.

If a CVE identifier exists for a particular vulnerability, the official CVE identifier SHALL be used. If no CVE exists for the software flaw, an alternate identifier MAY be used, but the user SHOULD seek to have a CVE identifier issued for the vulnerability. The process for submitting unpublished vulnerabilities and obtaining CVE identifiers is available from The MITRE Corporation via http://cve.mitre.org/cve/obtain_id.html.

NIST provides a CVE data feed to support dynamic and current vulnerability information and associated metadata (e.g., CVSS values). The current schema is available at http://nvd.nist.gov/download.cfm.

3.8 CVSS

The NIST CVE data feed, discussed in Section 3.7, is one source of CVSS base score and vector data that MAY be used by products to support additional use cases built on SCAP usage. In support of these additional use cases, CVSS base scores and vectors from this data feed MAY be used by products along with temporal, and environmental scores and vectors from other sources .

Additional information on CVSS use is available in NIST IR 7435, *The Common Vulnerability Scoring System (CVSS) and Its Applicability to Federal Agency Systems* (http://csrc.nist.gov/publications/PubsNISTIRs.html).

[17] http://nvd.nist.gov/cce.cfm

4. SCAP Processing Requirements and Recommendations

This section defines the processing requirements that tools MUST follow in order to correctly process SCAP 1.1 content. This section also provides recommendations that are not mandatory; organizations are encouraged to adopt them to promote stronger interoperability and greater consistency. The topics covered in this section are legacy support, SCAP content validation, the *<xccdf:Profile>* element, and check system usage. The end of the section covers result-related topics: SCAP result data streams, XCCDF results, OVAL results, and OCIL results.

4.1 Legacy Support

Products supporting SCAP 1.1 SHALL process SCAP 1.0 content as described under the SCAP 1.0 version of NIST SP 800-126.[18]

Products supporting OVAL SHALL support OVAL Definition documents written against OVAL versions 5.3, 5.4, 5.5, 5.6, 5.7, and 5.8.

Within the OVAL Language, constructs may be deprecated.[19] Deprecated constructs MUST be handled properly during OVAL Definition evaluation. Similar to the requirement to support previous minor versions of OVAL, this requirement will ensure that content that made use of these deprecated constructs continues to be supported in SCAP.

4.2 SCAP Content Validation

An SCAP implementation that can import SCAP content SHALL be capable of validating the content against the appropriate schemas and Schematron style sheets, detecting and reporting errors, and failing gracefully if there are errors.

4.3 The <xccdf:Profile> Element

If an *<xccdf:Profile>* element is not provided or selected, then profile processing SHALL be skipped and standard XCCDF benchmark processing rules SHALL apply.[20]

4.4 CPE Applicability Processing

When evaluating an *<xccdf:platform>* element in XCCDF content, it is necessary to evaluate machine state to determine the presence of a referenced CPE on the machine. CPEs referenced in an *<xccdf:platform>* element directly or by a *<cpe-lang:fact-ref>* contained within a referenced *<cpe-lang:platform-specification>* element SHALL be evaluated as follows:

1. The *<cpe_dict:cpe-item>* element data SHALL be located from the CPE dictionary data stream component in the same data stream with the *@name* attribute that is identical to the referenced CPE Name.

[18] http://csrc.nist.gov/publications/PubsSPs.html.
[19] The OVAL Language Deprecation policy is available here: http://oval.mitre.org/language/about/deprecation.html
[20] See NIST IR 7275r3, The XCCDF Specification version 1.1.4, p.36 section "Benchmark Processing Algorithm" for additional details (http://csrc.nist.gov/publications/nistir/ir7275r3/NISTIR-7275r3.pdf).

2. The *<cpe_dict:check>* element data associated with the identified *<cpe_dict:cpe-item>* element SHALL be evaluated using the referenced CPE inventory data stream component within the same data stream.

3. The result of evaluation SHALL be handled according to Section 4.7.3 of this document, with a result of "pass" indicating that the CPE Name was found on the machine.

4.5 Check System Usage

In XCCDF content, if multiple *<xccdf:check-content-ref>* elements are provided, then the following evaluation method SHALL be performed:

1. Evaluate each *<xccdf:check-content-ref>* element in the order that it appears in the *<xccdf:check>* element. The first resolvable *<xccdf:check-content-ref>* element SHALL be used to determine the *<xccdf:Rule>* status.

2. For each *<xccdf:check-content-ref>* element, a product will attempt to retrieve the document referenced by the *@href* attribute. If not resolvable, the next available *<xccdf:check-content-ref>* element SHALL be evaluated. If none of the *<xccdf:check-content-ref>* elements are resolvable, then the result of the rule evaluation SHALL be the XCCDF "unchecked" status and processing of the *<xccdf:Rule>* SHALL end. Please note that it is acceptable to map a remote URL to a local copy of the file in cases where remote access is not available, not allowed, or not practical.

3. Once a resolvable *<xccdf:check-content-ref>* element is found, then check system processing SHALL proceed. When evaluating a rule, an *<xccdf:rule-result/xccdf:message>* with the *@severity* attribute value of "info" SHALL be generated, indicating the *<xccdf:check-content-ref>* *@href* and *@name*, if provided.

Use of XCCDF check systems as specified in the *<xccdf:check>* element's *@system* attribute SHALL be restricted as follows:

1. SCAP scanning products SHALL implement the SCAP supported check systems that are required for the SCAP capability or capabilities that the products offer. The SCAP supported check systems are:

   i. OVAL check system. Use of the OVAL check system SHALL be indicated by the *http://oval.mitre.org/XMLSchema/oval-definitions-5* system identifier.

   ii. OCIL check system. Use of the OCIL check system SHALL be indicated by the *http://scap.nist.gov/schema/ocil/2* system identifier.

2. SCAP scanning tools MAY implement check systems that are not supported by SCAP.

3. Evaluation of an *<xccdf:check>* containing a reference to a non-SCAP check system SHALL produce an "unchecked" result if an SCAP scanning product does not implement the check system.

An *<xccdf:check-content-ref>* element may omit the *@name* attribute only for a patches up-to-date rule (see Section 3.2.6.4). When processing a patches-up-to-date rule, only OVAL patch class

definitions SHALL be evaluated; all other classes of definitions (e.g., inventory class definitions) SHALL NOT be evaluated.

4.6 SCAP Result Data Streams

An SCAP result data stream contains the results of the evaluation of one or more SCAP source data streams by an SCAP product. Correlation and aggregation products such as security awareness incident response tools may consume properly formatted SCAP result data streams to support organizational reporting requirements.

For its filenames, every SCAP result data stream SHALL use two common locator prefixes that are appended to the URL base of the deployed result file. The first locator prefix (a string followed by a hyphen) SHALL be associated with a specific result data stream. The first locator prefix SHALL be consistent between multiple evaluations of the same source content. The second locator prefix (a string followed by a hyphen) MAY be used to differentiate among similar result data streams.

Every SCAP result data stream component SHALL have a filename comprised of the first locator prefix, the second locator prefix, and the appropriate component suffix (as listed in Table 5), in that order. Each component SHALL use the element specified in Table 5 as its document element.

Table 5. SCAP Result Data Stream Naming Conventions

Component	Component Suffix	Document Element
XCCDF Benchmark	xccdf-res.xml	`<xccdf:Benchmark>` or `<xccdf:TestResults>`
OVAL Compliance	oval-res.xml	`<oval-def:oval_definitions>`
OVAL Patch	patches-res.xml	`<oval-def:oval_definitions>`
OVAL Vulnerability	oval-res.xml	`<oval-def:oval_definitions>`
OCIL Questionnaire	ocil-res.xml	`<ocil:ocil>`
CPE Inventory	cpe-oval-res.xml	`<oval-def:oval_definitions>`

4.7 XCCDF Results

Each XCCDF result data stream component SHALL comply with the XCCDF Results schema.

XCCDF test results SHALL be documented as the contents of an *<xccdf:TestResult>* element that either stands alone as the root of an XML document or is embedded as a child-element of an *<xccdf:Benchmark>* root element. In the former case, the *<xccdf:TestResults>* document requires an embedded *<xccdf:benchmark>* element that identifies the associated benchmark. In the latter case, the associated benchmark is the embedding benchmark; *<xccdf:benchmark>* elements SHALL be ignored in *<xccdf:TestResult>* elements that are embedded in their associated benchmark.

To be considered valid SCAP result content, the following conditions SHALL be met:

1. One or more *<xccdf:organization>* elements SHALL be provided to indicate the organizational units responsible for applying the checklist.

2. The *@start-time* and *@end-time* attributes SHALL be provided to indicate when the scan started and completed, respectively.

3. The *@test-system* attribute SHALL be provided with a CPE Name value indicating the product that evaluated the checklist.

4. If the *<xccdf:TestResult>* is the root XCCDF element, the *<xccdf:benchmark>* element's *@href* attribute SHALL be an absolute URL, NOT a relative URL.

5. Regarding the definition and use of *<xccdf:Profile>* elements:

    a. If no *<xccdf:Profile>* was selected, then the *<xccdf:Profile>* SHALL be omitted.

    b. When using a profile during the processing of XCCDF content, the test results SHALL embed an *<xccdf:profile>* element that contains the name of the utilized profile.

    c. Reported rule results SHALL include all selected rules within the specified Profile.

    d. Reported value-settings SHALL include all those values that are exported by the reported rules. The specific settings are those determined by the reported Profile.

6. The *<xccdf:identity>* element SHALL identify the security principal used to access rule evaluation on the target(s).

7. Each IP address associated with the *<xccdf:target>* SHALL be enumerated using the *<xccdf:target-address>* element.

8. The *<xccdf:rule-result>* elements SHALL report the result of the application of each selected rule against all specified targets.

    a. The *@idref* attribute of the *<xccdf:rule-result>* SHALL identify the selected rule.

    b. If an evaluated rule references a check system (e.g., OVAL, OCIL) that the SCAP implementation does not support, the implementation SHALL return a result of "notchecked" for each such rule.

    c. The *<xccdf:check/xccdf:check-content-ref>* element SHALL record the reference to the check system specific result file and check name within the result file using the *@href* and *@name* attributes, respectively. This approach provides traceability between XCCDF and check results.

9. Where applicable to the target system, each of the following *<xccdf:fact>* elements SHALL be provided:

Table 6. XCCDF Fact Descriptions

XCCDF Fact	Description of Use
urn:scap:fact:asset:identifier:mac	Ethernet media access control address
urn:scap:fact:asset:identifier:ipv4	Internet Protocol version 4 address
urn:scap:fact:asset:identifier:ipv6	Internet Protocol version 6 address
urn:scap:fact:asset:identifier:host_name	Host name of the asset, if assigned
urn:scap:fact:asset:identifier:fqdn	Fully qualified domain name
urn:scap:fact:asset:identifier:ein	Equipment identification number or other inventory tag number

urn:scap:fact:asset:identifier:guid	Globally unique identifier for the asset, if assigned
urn:scap:fact:asset:environmental_information:owning_organization	Organization that tracks the asset on its inventory
urn:scap:fact:asset:environmental_information:current_region	Geographic region where the asset is located
urn:scap:fact:asset:environmental_information:administration_unit	Name of the organization that does system administration for the asset

4.7.1 Assigning CVE Identifiers to Rule Results

The *<xccdf:rule-result>* element provides data indicating the result of assessing a system using the identified *<xccdf:Rule>* element. If the target *<xccdf:Rule>* identified by the <xccdf:*rule-result idref=””>* attribute has one or more *<ident>* elements with the "*http://cve.mitre.org"* or *"CVE"* system identifiers, then each *<xccdf:ident>* element SHALL also appear within the *<xccdf:rule-result>* element.

For example:

```
<xccdf:rule-result idref="java-upgrade-278" weight=”10.0”>
   <xccdf:result>pass</xccdf:result>
   …
   <xccdf:ident system="http://cve.mitre.org”>CVE-2006-0614</xccdf:ident>
   …
</xccdf:rule-result>
```

An *<xccdf:rule-result>* of "pass" SHALL indicate that the target platform satisfies all the conditions of the XCCDF rule and is unaffected by the vulnerability or exposure referenced by the CVE.

4.7.2 Assigning CCE Identifiers to Rule Results

The *<xccdf:rule-result>* element provides data indicating the result of assessing a system using the identified <xccdf:Rule> element. If the target *<xccdf:Rule>* identified by the *<xccdf:rule-result> @idref* attribute has one or more *<xccdf:ident>* elements with the "*http://cce.mitre.org"* or *"CCE"* system identifiers, then each *<xccdf:ident>* element SHALL also appear within the *<rule-result>* element. For example:

```
<xccdf:rule-result idref="minimum_password_length">
   <xccdf:result>pass</xccdf:result>
   …
   <xccdf:ident system="http://cce.mitre.org">CCE-2981-9</xccdf:ident>
   …
</xccdf:rule-result>
```

An *<xccdf:rule-result>* of "pass" SHALL indicate that the target platform complies with the configuration setting guidance expressed in the XCCDF rule.

4.7.3 Mapping OVAL Results to XCCDF Results

When evaluating an *<xccdf:Rule>* element that references an OVAL Definition, the *<xccdf:rule-result>* element SHALL be used to capture the result of this evaluation. This result SHALL be determined by evaluating the referenced OVAL Definition on a target host. The *<xccdf:result>* value recorded SHALL be mapped from the OVAL Definition Result produced during evaluation.

While the OVAL specification permits limiting result status reporting, SCAP-compliant content SHALL include full status reporting including Error, Unknown, Not Applicable, Not Evaluated, True, and False.

SCAP compliant processors that generate XCCDF *<xccdf:rule-result>* elements SHALL apply the mapping illustrated in Table 7 when deriving *<xccdf:Rule>* results from OVAL Definition processing. The corresponding *<xccdf:rule-result/xccdf:result>* value SHALL be recorded based on the *@class* of the OVAL Definition where applicable.

Table 7. Deriving XCCDF Rule Results from OVAL Definition Results

<table>
<tr><th colspan="2">OVAL Definition Result</th><th>XCCDF Rule Result</th></tr>
<tr><td colspan="2">error</td><td>error</td></tr>
<tr><td colspan="2">unknown</td><td>unknown</td></tr>
<tr><td colspan="2">not applicable</td><td>notapplicable</td></tr>
<tr><td colspan="2">not evaluated</td><td>notchecked</td></tr>
<tr><td colspan="2"><table><tr><th>Definition Class</th><th>Definition Result</th></tr><tr><td>compliance</td><td>true</td></tr><tr><td>vulnerability</td><td>false</td></tr><tr><td>inventory</td><td>true</td></tr><tr><td>patch</td><td>false</td></tr></table></td><td>pass</td></tr>
<tr><td colspan="2"><table><tr><th>Definition Class</th><th>Definition Result</th></tr><tr><td>compliance</td><td>false</td></tr><tr><td>vulnerability</td><td>true</td></tr><tr><td>inventory</td><td>false</td></tr><tr><td>patch</td><td>true</td></tr></table></td><td>fail</td></tr>
</table>

Some of the mappings in Table 7 may seem counterintuitive, but when viewed in the appropriate context the underlying logic is evident. For example, if an OVAL Definition of class "compliance" is processed and the XCCDF returns a result of "true", the product is conveying the fact that the system was found to be compliant with that check and therefore returns a "pass" result. A similar definition for a vulnerable condition will return results of "false" if that vulnerability was not found on the examined devices, resulting in a "pass" from the XCCDF rule.

If the *<xccdf:Rule>* under evaluation has an *<xccdf:check-content-ref>* element with the *@name* attribute omitted, then the result of each OVAL Definition SHALL be evaluated as a separate *<xccdf:rule-result>*. This will commonly occur for a *"security_patches_up_to_date"* check, as defined in Section 3.2.6.4. In this case the *<xccdf:rule-result/xccdf:check-content-ref>* SHALL identify the specific check result of each evaluated OVAL definition using the *@href* and *@name* attributes as described in Section 4.7.

4.8 OVAL Results

Each SCAP OVAL result data stream component SHALL use the *<oval-res:oval_results>*

element as the document element. Each OVAL result data stream component SHALL validate against version 5.8 of the OVAL Results schema[21] regardless of the version of the OVAL Definitions document that was evaluated.

An SCAP OVAL result data stream component SHALL include the results of every OVAL Definition used to generate the reported results.

In order to be SCAP compliant, an SCAP scanning product SHALL be able to produce both thin and full OVAL Results output as described below. The specific result output SHALL be configurable within the SCAP product.

In order to support SCAP instances where OVAL thin content (only the ID of the definition and the results) is preferred, SCAP products SHALL support all valid values for the *<oval-res:directives>* controlling the expected content of the results file.

To support the ability for results to be consumed by the appropriate product(s), data results SHALL be expressed as Single Machine Without System Characteristics, Single Machine With System Characteristics, or Single Machine With Thin Results as follows:

1. Single Machine Without System Characteristics – A single result file that includes all OVAL definitions evaluated and "full" results types as described in the ContentEnumeration element of the OVAL Results schema[22], without system characteristics.

   For this format, the values for the `<oval-res:directives>` element SHALL be:

   ```
   <oval-res:definition_true content="full" reported="true"/>
   <oval-res:definition_false content="full" reported="true"/>
   <oval-res:definition_unknown content="full" reported="true"/>
   <oval-res:definition_error content="full" reported="true"/>
   <oval-res:definition_not_evaluated content="full" reported="true"/>
   <oval-res:definition_not_applicable content="full" reported="true"/>
   ```

2. Single Machine With System Characteristics – A single result file that includes all OVAL definitions evaluated and "full" results types as described in the ContentEnumeration element of the OVAL Results schema and the System Characteristics of the target evaluated.

   For this format, the values for the `<oval-res:directives>` element SHALL be:

   ```
   <oval-res:definition_true content="full" reported="true"/>
   <oval-res:definition_false content="full" reported="true"/>
   <oval-res:definition_unknown content="full" reported="true"/>
   <oval-res:definition_error content="full" reported="true"/>
   <oval-res:definition_not_evaluated content="full" reported="true"/>
   <oval-res:definition_not_applicable content="full" reported="true"/>
   ```

   When creating the OVAL System Characteristics as defined by the `<oval-sc:oval_system_characteristics>` element, the `<oval-sc:collected_objects>` and `<oval-sc:system_data>` elements SHALL be provided.

[21] The OVAL schemas are described in detail at http://oval.mitre.org/language/about.
[22] The OVAL Results schema is described at http://oval.mitre.org/language/about/structure.html#results.

3. Single Machine With Thin Results – A single result file that includes all OVAL definitions evaluated and "thin" results types as described in the OVAL Results schema. A value of "thin" means only the minimal amount of information will be provided.

   For this format, the values for the `<oval-res:directives>` element SHALL be:

```
<oval-res:definition_true content="thin" reported="true"/>
<oval-res:definition_false content="thin" reported="true"/>
<oval-res:definition_unknown content="thin" reported="true"/>
<oval-res:definition_error content="thin" reported="true"/>
<oval-res:definition_not_evaluated content="thin" reported="true"/>
<oval-res:definition_not_applicable content="thin" reported="true"/>
```

4.9 OCIL Results

An SCAP OCIL result data stream component SHALL include the results of every OCIL questionnaire, test_action, and question used to generate the reported results.

5. Data Stream Content Types

This section discusses additional requirements for four common types of SCAP content: compliance checking, vulnerability scanning, inventory scanning, and OVAL-only scanning.

5.1 Compliance Checking

SCAP content can be used to compare system characteristics and settings against an SCAP-expressed checklist in an automated fashion. This can verify that operating systems and applications comply with security checklists and identify any deviations from those checklists.

The SCAP source data stream component that MUST be included for compliance checking is the XCCDF Benchmark, which expresses the checklist. Each rule in the XCCDF Benchmark SHALL reference one of the following:

- An OVAL compliance definition. This definition SHALL be contained in an OVAL Compliance component, which holds the definitions of the compliance checks used by the checklist. The OVAL Compliance component SHALL have at least one OVAL definition of class compliance, MAY have one or more additional OVAL definitions of classes compliance and/or inventory, and SHALL NOT have any other classes of OVAL definitions. An XCCDF Benchmark's rules MAY reference one or more OVAL compliance definitions in an OVAL Compliance component.
- An OCIL questionnaire. This questionnaire SHALL be contained in an OCIL Questionnaire component, which holds the questionnaires that collect information that OVAL is not being used to collect, such as posing questions to users or harvesting configuration information from an existing database. An XCCDF Benchmark's rules MAY reference one or more OCIL questionnaires in an OCIL Questionnaire component.
- An OVAL Patch component. The OVAL Patch component holds definitions for patch compliance checks. These checks may be needed if an organization includes patch verification in its compliance activities. The OVAL Patch component SHALL have at least one OVAL definition of class patch, MAY have one or more additional OVAL definitions of classes compliance and/or inventory, and SHALL NOT have any other classes of OVAL definitions. An XCCDF Benchmark MAY reference an OVAL Patch component through a patches up-to-date rule in a manner consistent with Section 3.2.6.4.

Each XCCDF Benchmark SHALL have at least one rule that references either an OVAL compliance definition in the OVAL Compliance component or an OCIL questionnaire in the OCIL Questionnaire component.

All OVAL Compliance, OCIL Questionnaire, and OVAL Patch components referenced by the XCCDF Benchmark SHALL be included in the SCAP source data stream.

If the XCCDF Benchmark component references any CPE names, then the SCAP source data stream MUST include the following components, in addition to those already mentioned:

- CPE Dictionary: specifies the products or platforms of interest.
- CPE Inventory: contains the technical procedures for determining whether or not a specific target asset has a product or platform of interest. The CPE Inventory component SHALL have one or more OVAL definitions of class inventory and SHALL NOT have any other classes of OVAL definitions.

5.2 Vulnerability Scanning

SCAP content can be used to scan operating systems and applications to look for known software flaws that introduce security exposures. The content enables consistent detection and reporting of these flaws.

The SCAP source data stream component that MUST be included for vulnerability scanning is the XCCDF Benchmark, which expresses the checklist of the flaws to be checked for. Each rule in the XCCDF Benchmark SHALL reference one of the following:

- An OVAL vulnerability definition. This definition SHALL be contained in an OVAL Vulnerability component, which holds the definitions of the vulnerability checks used by the checklist. The OVAL Vulnerability component SHALL have at least one OVAL definition of class vulnerability, MAY have one or more additional OVAL definitions of classes vulnerability and/or inventory, and SHALL NOT have any other classes of OVAL definitions. An XCCDF Benchmark's rules MAY reference one or more OVAL vulnerability definitions in an OVAL Vulnerability component.
- An OCIL questionnaire. This questionnaire SHALL be contained in an OCIL Questionnaire component, which holds the questionnaires that collect information that OVAL is not being used to collect, such as giving a system administrator step-by-step directions for manually examining a system for a vulnerability that cannot be detected with OVAL, and then collecting information on the results of that manual examination. An XCCDF Benchmark's rules MAY reference one or more OCIL questionnaires in an OCIL Questionnaire component.
- An OVAL Patch component. The OVAL Patch component holds definitions for patch compliance checks. These checks may be needed if an organization includes patch verification in its vulnerability scanning activities. The OVAL Patch component SHALL have at least one OVAL definition of class patch, MAY have one or more additional OVAL definitions of classes compliance and/or inventory, and SHALL NOT have any other classes of OVAL definitions. An XCCDF Benchmark MAY reference an OVAL Patch component through a patches up-to-date rule in a manner consistent with Section 3.2.6.4.

Each XCCDF Benchmark SHALL have at least one rule that references either an OVAL vulnerability definition in the OVAL Vulnerability component or an OCIL questionnaire in the OCIL Questionnaire component.

All OVAL Vulnerability, OCIL Questionnaire, and OVAL Patch components referenced by the XCCDF Benchmark SHALL be included in the SCAP source data stream.

If the XCCDF Benchmark component references any CPE names, then the SCAP source data stream MUST include the following components, in addition to those already mentioned:

- CPE Dictionary: specifies the products or platforms of interest.
- CPE Inventory: contains the technical procedures for determining whether or not a specific target asset has a product or platform of interest. The CPE Inventory component SHALL have one or more OVAL definitions of class inventory and SHALL NOT have any other classes of OVAL definitions.

5.3 Inventory Scanning

SCAP content can be used to collect information on the software installed on systems. One example of how this could be used is to verify that a group of systems all have required security software programs installed. This could help verify compliance with technical security control requirements. Another

example is to collect software inventory data on devices that are not directly connected to the enterprise network, such as smart phones.

The SCAP source data stream components that MUST be included for inventory scanning are:

- XCCDF Benchmark: references the CPE Inventory and captures the results of the inventory. Each rule in the XCCDF Benchmark SHALL reference an OVAL inventory definition in the CPE Inventory component.
- CPE Inventory: contains the technical procedures for determining whether or not a specific target asset has a product or platform of interest. The CPE Inventory component SHALL have one or more OVAL definitions of class inventory and SHALL NOT have any other classes of OVAL definitions.

5.4 OVAL-Only Scanning

OVAL content can be used on its own, without XCCDF Benchmarks or other components, to perform scanning. The only SCAP source data stream component that MUST be included is an OVAL component that maps to the desired definition classes (e.g., compliance class for configuration setting checks, inventory class for asset checks, patch class for patch presence checks, vulnerability class for software flaw vulnerability presence checks). The mapping SHALL correspond to the mappings defined in Section 3.2.6.2.

Appendix A—Acronyms and Abbreviations

Selected acronyms and abbreviations used in the guide are defined below.

API	Application Programming Interface
CCE	Common Configuration Enumeration
CPE	Common Platform Enumeration
CVE	Common Vulnerabilities and Exposures
CVSS	Common Vulnerability Scoring System
DHS	Department of Homeland Security
DoD	Department of Defense
FDCC	Federal Desktop Core Configuration
FISMA	Federal Information Security Management Act
IR	Interagency Report
IT	Information Technology
ITL	Information Technology Laboratory
NIST	National Institute of Standards and Technology
NSA	National Security Agency
NVD	National Vulnerability Database
OCIL	Open Checklist Interactive Language
OMB	Office of Management and Budget
OS	Operating System
OVAL	Open Vulnerability and Assessment Language
PCI	Payment Card Industry
RFC	Request for Comments
SCAP	Security Content Automation Protocol
SP	Service Pack
SP	Special Publication
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
XCCDF	eXtensible Configuration Checklist Description Format
XML	eXtensible Markup Language

Appendix B—Normative References

The following normative references are pointers to the specifications, schema, dictionaries, and other information that are required to implement the SCAP 1.1 components:

[CCE]	CCE specification and description	http://scap.nist.gov/revision/1.1/index.html#cce
[CPE]	CPE specification and description	http://scap.nist.gov/revision/1.1/index.html#cpe
[CVE]	CVE specification and description	http://scap.nist.gov/revision/1.1/index.html#cve
[CVSS]	CVSS specification and description	http://scap.nist.gov/revision/1.1/index.html#cvss
[OCIL]	OCIL specification and description	http://scap.nist.gov/revision/1.1/index.html#ocil
[OVAL]	OVAL specification and description	http://scap.nist.gov/revision/1.1/index.html#oval
[XCCDF]	XCCDF specification and description	http://scap.nist.gov/revision/1.1/index.html#xccdf