SECURITY CONTENT AUTOMATION PROTOCOL: IS IT BENEFICIAL?

John J. Rayborn

East Carolina University

Author Note

This paper was prepared for course ICTN 6823, Information Security Management, taught by Dr. Phil Lunsford.

Abstract

The intent of this paper is to provide a discussion on what the Security Content Automation Protocol (SCAP) is, how organizations can utilize SCAP, and why SCAP is important to Information Security Management. Further, this paper will also look to explain the components and processes that make up SCAP, an overview to the process for vendor product's to receive SCAP validation, and how organizations look to ensure systems on the enterprise networks are compliant with Federal Information Security Management Act (FISMA) of 2002 requirements through SCAP tools and methodology usage. This paper is meant to be informational as well as educational for awareness and discussion purposes.

Trademark Information

Contents

Introduction

Even with the ongoing efforts of individuals looking to access corporate infrastructure and resources with BYOD systems, the use of a standard system has become inherently the norm for many organizations. When organizations, of various sizes and industries, look to utilize a standard system for use on the enterprise network, what does this really mean? Is a common image being provided to the hardware vendor prior to the order being placed? Or perhaps, there's already infrastructure in place that allows the organization to deploy this standard image to end-users. Whatever the case may be, more than likely, the system image is out of compliance and will require software updates and security patches to be considered current once it's placed on the network. The questions then become; what set of standards are systems compliant with, and how can an organization tell whether systems are compliant with these standards? This is where the Security Content Automation Protocol (SCAP), pronounced (ess-cap), can provide assistance as a component of an overall Information Security Management process.

This paper will provide an overview as to what SCAP is for discussion purposes. We will then look to see how SCAP came about and how it can be used by various organizations. A review of the specifications utilized by SCAP will be provided as well. Finally, we'll take a brief look at some of the components that entail SCAP and the validation process taken by third-party vendor products.

What is the Security Content Automation Protocol (SCAP)?

Before we can discuss the importance of SCAP, we first need to define what SCAP is. There are two main components that make up SCAP; first of which is the definition of the protocol itself, "The Security Content Automation Protocol is a suite of six specifications that standardize the format and nomenclature by which security software products communicate information about publicly known software flaws, and security configurations annotated with common identifiers and embedded in XML."[2] Second, is related to the SCAP content which contains reference data pertaining to the software flaws and security configurations mentioned previously. [2][3] The following six specifications are currently being utilized by SCAP:

- eXtensible Configuration Checklist Description Format (XCCDF)
- Open Vulnerability and Assessment Language (OVAL)
- Common Platform Enumeration (CPE)
- Common Configuration Enumeration (CCE)
- Common Vulnerability Scoring System (CVSS)
- Common Vulnerabilities and Exposures (CVE)

[2][3][16][19]

NOTE: While brief definitions of each of these specifications will be provided later, the XCCDF and OVAL specifications will be discussed in further detail for the purpose of this paper.

SCAP is one of the components that make up the overall Federal Information Security Management Act (FISMA) project by providing a standardized security configuration through SCAP content. Another component is provided through the National Institute for Standards and Technology (NIST) Special Publication (SP) 800-53 with the control framework to be utilized with SCAP content. The National Vulnerability Database (NVD) is the repository created by NIST for U.S. government usage, which contains standards based vulnerability management data utilizing SCAP. [5] Together with the NVD, these components provide organizations within the Federal government the capability to determine the current state of compliance of systems on their networks according to FISMA, Federal Desktop Core Configuration (FDCC), and even United States Configuration Government Baseline (USGCB) requirements. While the most current version available of SCAP is 1.2, the most current effective version is 1.0. [18] For the purposes of this paper, the effective version of 1.0 will be used for reference.

Why is SCAP Important?

While SCAP was developed to provide a manner in which Federal organizations can show compliance with mandated security compliance requirements created under the FISMA of 2002, there is a potential for even non-Federal organizations to benefit from what SCAP accomplishes. [2][3] SCAP provides the capability to utilize various third-party products to determine whether systems of an organization are in compliance based on SCAP content. Previous to SCAP, the process of accomplishing determinations of vulnerability states of systems was time consuming and manpower intensive, which provided the opportunity for errors and interpretations of the requirements trying to be met. [16] Utilizing SCAP allows an organization to map low-level technical details of a system's security posture and have them linked to the high-level security requirements from mandates such as FISMA. [16]

Since SCAP content is publicly available for use, it isn't necessarily specific to Federal agencies and organizations. As private sector organizations are being targeted more by attackers, there is an inherent need to determine the security state of these systems. Even with various mandates such as Health Insurance Portability and Accountability Act (HIPAA), Sarbanes-Oxley (SOX), and others such as ISO 27001 and DoD 8500 series [16] – there has been an increase in organizations looking to utilize more stringent security requirements as those mandated by Defense Information Systems Agency (DISA) Security Technical Implementation Guide (STIG)s.

SCAP Specifications.

As noted earlier in this paper, there are currently six specifications currently being utilized by SCAP which can be broken down into three categories. One category is based on the languages for specifying checklists, generating checklist reports, and specifying the low-level testing procedures used by these checklists. The second category entails the enumerations that include nomenclatures and dictionaries for security and product-related information. Finally, the third category provides for vulnerability measurement and scoring systems which generates scores based on the characteristics being measured as vulnerabilities. [2][3]

Extensible Configuration Checklist Description Format (XCCDF)

"XCCDF is a specification language for writing security checklists, benchmarks, and related kinds of documents." [10] Is based on XML in the creation of security configuration rules for operating systems and application platforms. [3][10] [19]

Open Vulnerability and Assessment Language (OVAL)

OVAL is international in scope and free for public use which uses an information security community effort to standardize how to assess and report upon the machine state of computer systems. [6] XML specification that provides technical details on configuration issues, software flaws, and patches. [3][19] OVAL standardizes the three steps of the assessment process by accomplishing the testing, analyzing, and reporting the results of configuration information against the system in question. [6][16]

Common Platform Enumeration (CPE)

"CPE is a structured naming scheme for information technology systems, software, and packages." [9] CPE provides a dictionary that is available for public use and is potentially updated on a nightly basis as modifications or names are added. [19]

Common Configuration Enumeration (CCE)

"Provides unique identifiers to system configuration issues in order to facilitate fast and accurate correlation of configuration data across multiple information sources and tools." [8] The CCE identifiers were being maintained by MITRE and is currently going through a transition for NIST to take over as the source repository for CCE lists.

Common Vulnerability Scoring System (CVSS)

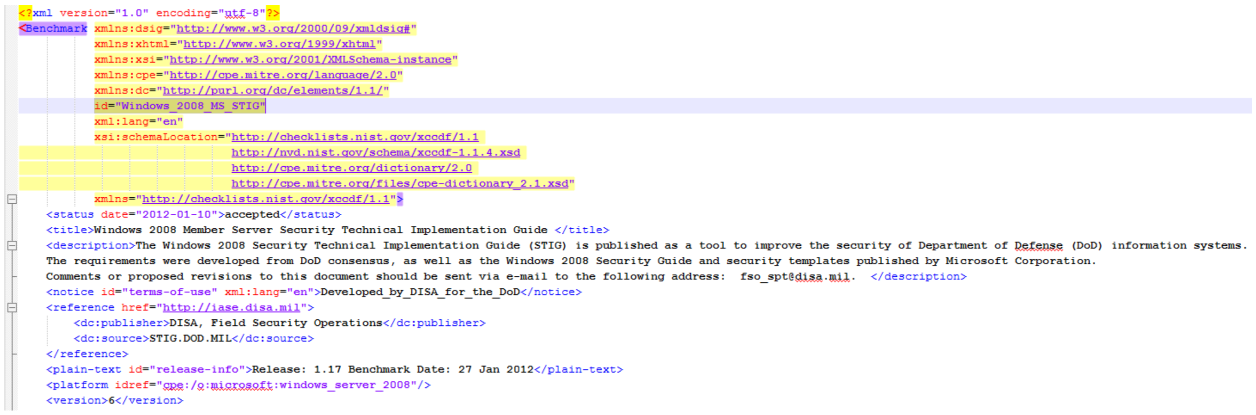
Comprises of a method for both classifying characteristics of software flaws and assigning severity scores based on these characteristics. [3][2][19] CVSS provides three metric groups in determining a vulnerability severity score: Basic, Temporal, and Environmental. Basic provides a generic score based on the intrinsic characteristics of the vulnerability being determined. Temporal provides an adjustment to the base score determined by the external factors of the vulnerability that changes over time. Environmental provides the severity of the vulnerability based on the context of the organization's operating environment. [2] CVSS provides a scoring scale of 0 to 10, with 10 indicating the highest severity possible for a vulnerability.

Common Vulnerabilities and Exposures (CVE)

CVE is international in scope and free for public use while also providing a dictionary of publicly known information security vulnerabilities and exposures. [7] CVE utilizes a common naming convention which allows the capability of sharing data within and among organizations which also allows for integration from various third-party services and tools. [2][19] CVE entries consist of the following information: unique name, short description of the vulnerability, and references to public advisories specific to the vulnerability.

How Does SCAP Work?

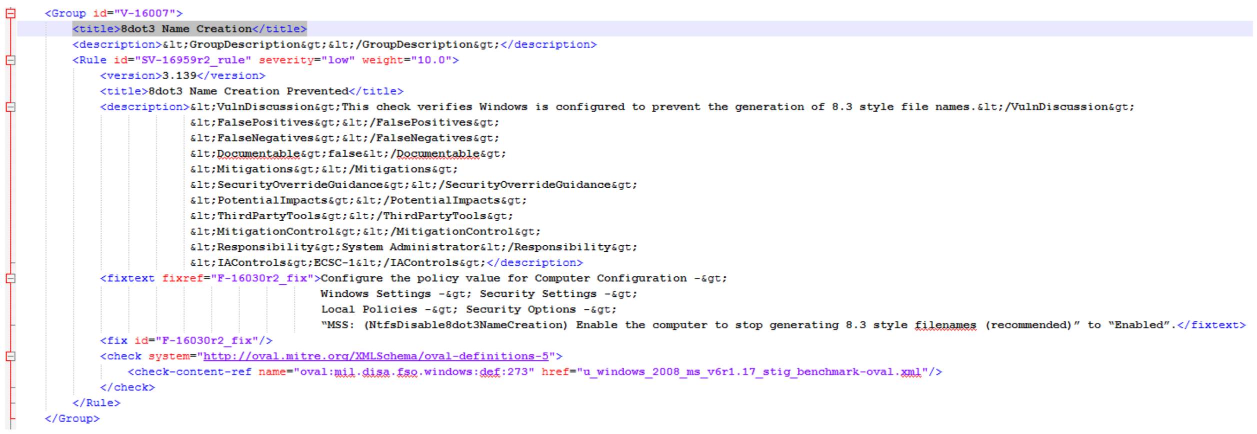
Now that we have a better understanding of what SCAP is and its importance to FISMA, we can now look at how SCAP works. For the purpose of this paper, we will review the metadata within a DISA STIG configuration specific to Windows Server 2008 Member Servers. This zip file contains several XML files based on XCCDF and OVAL formats. Below is the header section of a XCCDF file specific to a DISA STIG for a Windows Member Server 2008 system. [11]



```
<?xml version="1.0" encoding="utf-8"?>
<Benchmark xmlns:dsig="http://www.w3.org/2000/09/xmldsig#"
           xmlns:xhtml="http://www.w3.org/1999/xhtml"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xmlns:cpe="http://cpe.mitre.org/language/2.0"
           xmlns:dc="http://purl.org/dc/elements/1.1/"
           id="Windows_2008_MS_STIG"
           xml:lang="en"
           xsi:schemaLocation="http://checklists.nist.gov/xccdf/1.1
                               http://nvd.nist.gov/schema/xccdf-1.1.4.xsd
                               http://cpe.mitre.org/dictionary/2.0
                               http://cpe.mitre.org/files/cpe-dictionary_2.1.xsd"
           xmlns="http://checklists.nist.gov/xccdf/1.1">
  <status date="2012-01-10">accepted</status>
  <title>Windows 2008 Member Server Security Technical Implementation Guide </title>
  <description>The Windows 2008 Security Technical Implementation Guide (STIG) is published as a tool to improve the security of Department of Defense (DoD) information systems.
  The requirements were developed from DoD consensus, as well as the Windows 2008 Security Guide and security templates published by Microsoft Corporation.
  Comments or proposed revisions to this document should be sent via e-mail to the following address:  fso_spt@disa.mil.  </description>
  <notice id="terms-of-use" xml:lang="en">Developed_by_DISA_for_the_DoD</notice>
  <reference href="http://iase.disa.mil">
     <dc:publisher>DISA, Field Security Operations</dc:publisher>
     <dc:source>STIG.DOD.MIL</dc:source>
  </reference>
  <plain-text id="release-info">Release: 1.17 Benchmark Date: 27 Jan 2012</plain-text>
  <platform idref="cpe:/o:microsoft:windows_server_2008"/>
  <version>6</version>
```

Figure 1. XCCDF Header Example.

From the above screenshot, we can ascertain quite a bit of information specific to the XCCDF format. We can tell from the metadata due to the id that this benchmark is specific to the Windows 2008 Member Server STIG provided by DISA. [11] We are also able to ascertain the release version and date information as well. This is useful in determining whether the latest version of this file is being used for compliance validation purposes. Below shows metadata specific to a particular check (in this case 8dot3 Name Creation) being accomplished as a part of the Windows 2008 Member Server STIG. [11]

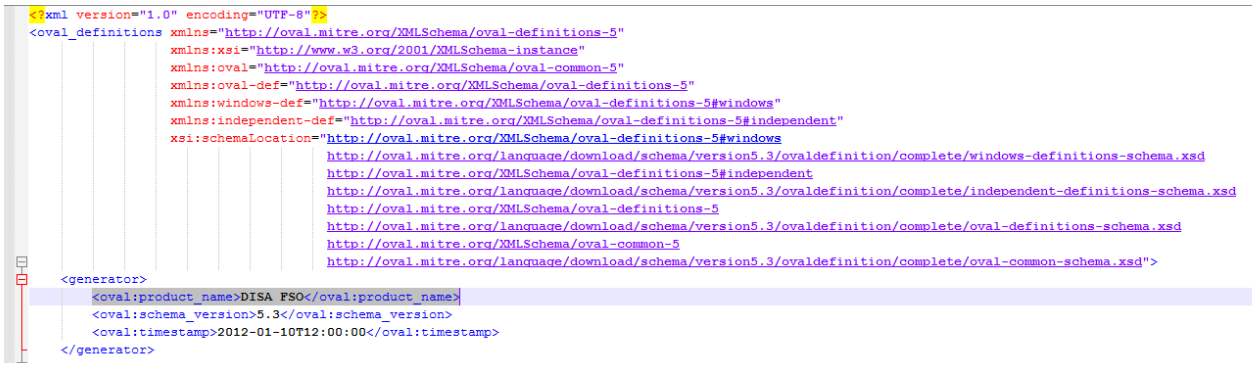


```
<Group id="V-16007">
    <title>8dot3 Name Creation</title>
    <description>&lt;GroupDescription&gt;&lt;/GroupDescription&gt;</description>
    <Rule id="SV-16959r2_rule" severity="low" weight="10.0">
        <version>3.139</version>
        <title>8dot3 Name Creation Prevented</title>
        <description>&lt;VulnDiscussion&gt;This check verifies Windows is configured to prevent the generation of 8.3 style file names.&lt;/VulnDiscussion&gt;
                    &lt;FalsePositives&gt;&lt;/FalsePositives&gt;
                    &lt;FalseNegatives&gt;&lt;/FalseNegatives&gt;
                    &lt;Documentable&gt;false&lt;/Documentable&gt;
                    &lt;Mitigations&gt;&lt;/Mitigations&gt;
                    &lt;SecurityOverrideGuidance&gt;&lt;/SecurityOverrideGuidance&gt;
                    &lt;PotentialImpacts&gt;&lt;/PotentialImpacts&gt;
                    &lt;ThirdPartyTools&gt;&lt;/ThirdPartyTools&gt;
                    &lt;MitigationControl&gt;&lt;/MitigationControl&gt;
                    &lt;Responsibility&gt;System Administrator&lt;/Responsibility&gt;
                    &lt;IAControls&gt;ECSC-1&lt;/IAControls&gt;</description>
        <fixtext fixref="F-16030r2_fix">Configure the policy value for Computer Configuration -&gt;
                                  Windows Settings -&gt; Security Settings -&gt;
                                  Local Policies -&gt; Security Options -&gt;
                                  "MSS: (NtfsDisable8dot3NameCreation) Enable the computer to stop generating 8.3 style filenames (recommended)" to "Enabled".</fixtext>
        <fix id="F-16030r2_fix"/>
        <check system="http://oval.mitre.org/XMLSchema/oval-definitions-5">
            <check-content-ref name="oval:mil.disa.fso.windows:def:273" href="u_windows_2008_ms_v6r1.17_stig_benchmark-oval.xml"/>
        </check>
    </Rule>
</Group>
```

Figure 2. XCCDF Check Example.

While the XCCDF format is being used, you can see that there are references to OVAL within this validation grouping. Specifically, the ***href="u_windows_2008_ms_v6r1.17_stig_benchmark-oval.xml"*** points to another file provided with the XCCDF file by DISA for validation purposes of this particular compliance check.[11] Another nice thing provided within the metadata is a recommended setting that would determine what a Windows Server 2008 Member Server in compliance with this check would be "Enabled" (However, in reading the verbiage for the compliance check, by configuring the item as "Enabled" actually disables the ability to create 8dot3 filenames). Also note in the check-content-ref line of the metadata there is a reference to ***name="oval:mil.disa.fso.windows:def:273"***, this will be pertinent when we review the OVAL file next.

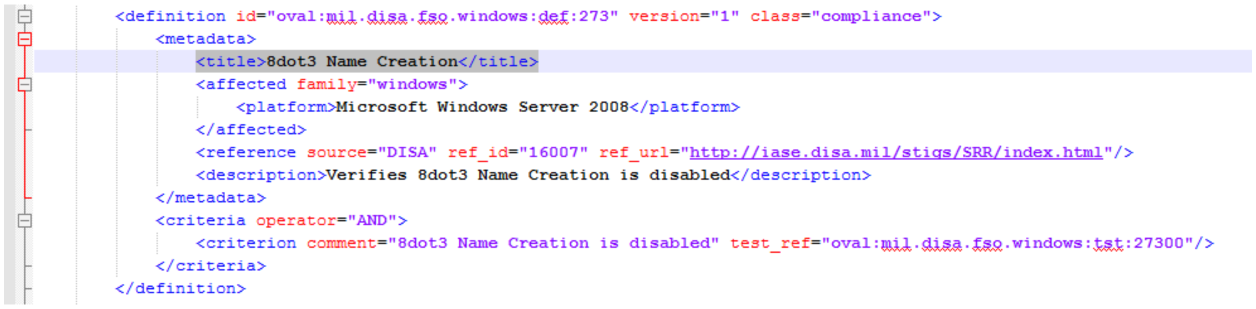
Below is the header section of the OVAL file specific to the same DISA STIG for a Windows Member Server 2008 system that was provided for XCCDF. [11]



```xml
<?xml version="1.0" encoding="UTF-8"?>
<oval_definitions xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5"
                  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xmlns:oval="http://oval.mitre.org/XMLSchema/oval-common-5"
                  xmlns:oval-def="http://oval.mitre.org/XMLSchema/oval-definitions-5"
                  xmlns:windows-def="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows"
                  xmlns:independent-def="http://oval.mitre.org/XMLSchema/oval-definitions-5#independent"
                  xsi:schemaLocation="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows
                                      http://oval.mitre.org/language/download/schema/version5.3/ovaldefinition/complete/windows-definitions-schema.xsd
                                      http://oval.mitre.org/XMLSchema/oval-definitions-5#independent
                                      http://oval.mitre.org/language/download/schema/version5.3/ovaldefinition/complete/independent-definitions-schema.xsd
                                      http://oval.mitre.org/XMLSchema/oval-definitions-5
                                      http://oval.mitre.org/language/download/schema/version5.3/ovaldefinition/complete/oval-definitions-schema.xsd
                                      http://oval.mitre.org/XMLSchema/oval-common-5
                                      http://oval.mitre.org/language/download/schema/version5.3/ovaldefinition/complete/oval-common-schema.xsd">
    <generator>
        <oval:product_name>DISA FSO</oval:product_name>
        <oval:schema_version>5.3</oval:schema_version>
        <oval:timestamp>2012-01-10T12:00:00</oval:timestamp>
    </generator>
```

Figure 3. OVAL Header Example.

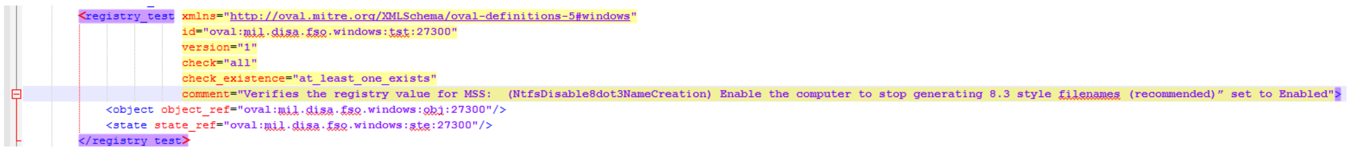
We can see that there's some differences between the header information contained in the XCCDF file and the OVAL file from the above screenshot.



```xml
<definition id="oval:mil.disa.fso.windows:def:273" version="1" class="compliance">
    <metadata>
        <title>8dot3 Name Creation</title>
        <affected family="windows">
            <platform>Microsoft Windows Server 2008</platform>
        </affected>
        <reference source="DISA" ref_id="16007" ref_url="http://iase.disa.mil/stigs/SRR/index.html"/>
        <description>Verifies 8dot3 Name Creation is disabled</description>
    </metadata>
    <criteria operator="AND">
        <criterion comment="8dot3 Name Creation is disabled" test_ref="oval:mil.disa.fso.windows:tst:27300"/>
    </criteria>
</definition>
```

Figure 4. OVAL Check Example Part 1.

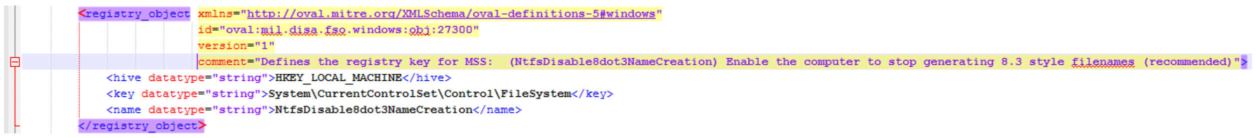
This portion of the metadata within the OVAL file provides information as to what the compliance check will be looking for as well as the definition ***id="oval:mil.disa.fso.windows:def:273"***.[11] This should be familiar from the XCCDF file discussed earlier. There are three instances related to the 27300 test to be accomplished as part of this compliance check.



```xml
<registry_test xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows"
               id="oval:mil.disa.fso.windows:tst:27300"
               version="1"
               check="all"
               check_existence="at_least_one_exists"
               comment="Verifies the registry value for MSS:  (NtfsDisable8dot3NameCreation) Enable the computer to stop generating 8.3 style filenames (recommended)" set to Enabled">
    <object object_ref="oval:mil.disa.fso.windows:obj:27300"/>
    <state state_ref="oval:mil.disa.fso.windows:ste:27300"/>
</registry_test>
```

Figure 5. OVAL Check Example Part 2.

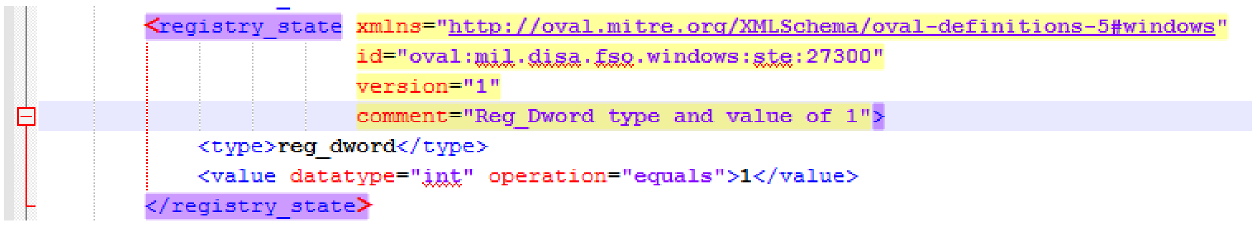
Figure 5 shows the metadata information that is going to be tested for compliance, this particular check is to be accomplished against the registry.



```
<registry_object xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows"
                 id="oval:mil.disa.fso.windows:obj:27300"
                 version="1"
                 comment="Defines the registry key for MSS:  (NtfsDisable8dot3NameCreation) Enable the computer to stop generating 8.3 style filenames (recommended)">
    <hive datatype="string">HKEY_LOCAL_MACHINE</hive>
    <key datatype="string">System\CurrentControlSet\Control\FileSystem</key>
    <name datatype="string">NtfsDisable8dot3NameCreation</name>
</registry_object>
```

Figure 6. OVAL Check Example Part 3.

The portion in Figure 6 shows the metadata provides the registry location that will be tested for compliance as part of this check.



```
<registry_state xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5#windows"
                id="oval:mil.disa.fso.windows:ste:27300"
                version="1"
                comment="Reg_Dword type and value of 1">
    <type>reg_dword</type>
    <value datatype="int" operation="equals">1</value>
</registry_state>
```

Figure 7. OVAL Check Example Part 4.

Figure 7 provides the information within the metadata as to what the expected registry value should be in order for the system to be compliant with this check. In this case, the REG_DWORD should equal “1” and be of datatype “int”. [11]

Now that we have a better understanding on the format of the SCAP content within both an XCCDF and OVAL files for a specific check, we can review the components that make SCAP a usable capability on an organization’s network.

What Are the Components of SCAP?

You may be asking yourself, “Great, I now have a better understanding of SCAP content but, how do I accomplish determining the security state of those systems we want to check against this content?” That’s where the third-party vendors enter the overall process in providing SCAP validated products. While an overview will be provided on the validation process a little later in this paper, we will now a background on the various components that vendors will look to become validated on for public usage.

As of SCAP 1.0, there were five different components available for SCAP validation for vendors, which are listed below:

- FDCC Scanner – this component provides the capability to audit and assess target systems based on compliance with FDCC requirements. [17]

- USGCB Scanner – similar to the FDCC Scanner, this component provides the capability to audit and assess a targeted systems based on compliance with USGCB content. [17]

- Authenticated Configuration Scanner (ACS) – determines compliance through the capability in auditing and assessing a target system using a defined set of configuration requirements utilizing target system logon privileges. [17][19]

- Authenticated Vulnerability and Patch Scanner – provides the capability to scan a target system in order to identify known vulnerabilities and evaluate software patch status. Ascertains compliance of a system by using a defined path policy utilizing target system logon privileges. [17]

- Unauthenticated Vulnerability Scanner – provides the capability in determining the presence of known vulnerabilities by evaluating a target system over the network without utilizing system logon privileges. [17]

Although it is not required for all of the components described above to be implemented in an organization’s environment for use; the utilization of one, some, or all is a possibility depending on the mandate compliance requirements of the organization. The main concept to be realized is that there are many portions of accomplishing a SCAP solution. From the SCAP content of what compliance data will be used specific to the organizations requirements (FISMA, DISA, etc…), to the vulnerability data returned from systems for analysis, to the reports generated in providing the overall state of the target systems on the network. The combination of all of these components make up a SCAP solution.

SCAP Validated Products

A rigorous and potentially long testing process is accomplished for third party vendors wanting to achieve SCAP validation for their products to be eligible as component of a SCAP solution. The validation program designed to test a product's ability in the use of SCAP features and functionality and SCAP component standards. [17] Under this program, independent laboratory tests are accomplished on the products looking to attain SCAP validation. These independent laboratories are accredited by the NIST National Voluntary Laboratory Accreditation Program (NVLAP) and are defined in NIST Handbook 150 and NIST Handbook 150-17. [17] Vendor products that are awarded validation are publicly posted at the NIST SCAP Validated Tools web page (http://nvd.nist.gov/scapproducts.cfm). [17] This page provides both a list of the SCAP 1.2 validated products, a link to the validated products for SCAP 1.0 (which expired on December 31, 2013), as well as a list of the NVLAP testing laboratories and the accredited testing scopes each can accomplish specific to SCAP.

A vendor's product may be validated for conformity to one or more of the six SCAP component specifications or conforming to a particular SCAP capability. SCAP capabilities are the manner(s) or ways that a product would use SCAP and are not necessarily related to specific product types. SCAP capabilities and can be classified under the various components listed above, such as "authenticated vulnerability scanner" or "authenticated vulnerability and patch scanner", etc... [16] To ensure that vendor products stay in-line with emerging SCAP technologies, the validation period of a vendor's product expires after one year unless a revalidation is accomplished. Validation also applies to the actual version of the product that was being tested. “Although subsequent versions of validated products may retain SCAP functionality tested with previous versions, NIST recommends that organizations acquire the most recent version of SCAP-validated products to receive the greatest SCAP functionality and the most capable version of the vendor’s product.” [16]

While there are currently only three product vendors that have achieved SCAP validation for version 1.2, there are several others that are still being reviewed by independent laboratories at the time this paper was being written. There were well over 30 product vendors that received SCAP validation for one or more of the SCAP components listed in the previous section for version 1.0.

Conclusion

As organizations become an even more heterogeneous environment with the addition of BYOD systems as well as enterprise workstations and servers of various operating systems and software applications, it becomes apparent as to the need to maintain a secure posture for the organization and its users. This is even more so the case for Federal agencies and organizations that have to meet security requirements of mandates such as FISMA of 2002 as part of an overall Information Security Management program. As time consuming as security configuration checks take in being accomplished by IT personnel, there has to be a better way to verify a systems security configuration; that way is through the utilization of SCAP.

In my experience with SCAP, there are various tools to assist in the process of accomplishing a SCAP solution for almost any environment. Like most implementations of first iteration projects, the training up-front of personnel is key as the processes put in place are potentially much different from the way Information Security validation business has been accomplished previously. Once the process has been learned and engrained into the organization, the operations and maintenance of the SCAP process will become second nature.

With the constantly increasing amount of attacks being accomplished toward networks and other organizational assets, SCAP provides an excellent manner in which a Federal organization can determine the security state of targeted systems based on FISMA requirements. As discussed in this paper, while the intent for SCAP was specific to Federal agencies and organizations for reporting purposes to the Office of Management and Budget (OMB), even private sector companies can benefit through its implementation and usage. Having SCAP content publicly available, for instance with DISA STIGs, and organizations being able to create their own SCAP content based on the six specifications mentioned in this document is a huge benefit. By SCAP providing this capability, the use of this standard provides the ability to share SCAP content with a variety of organizations and industries.

I think a potential downfall of SCAP is to ensure the consistent use and re-usage by various private and public sector entities as well as constant feedback to NIST in making the SCAP program better. Since SCAP is based on community input and feedback through the guidance provided by NIST, this communication needs to be maintained if SCAP is going to have long-term success in accomplishing what it set out to do – provide a better and faster way in determining the security state of targeted systems.

References

[1] National Institute of Standards and Technology. (January 24, 2014). The Security Content Automation Protocol (SCAP). Security Content Automation Protocol. Retrieved on July 6, 2014, from http://scap.nist.gov/

*[2] Quinn, S., Waltermire, D., Johnson, C., Scarfone, K., Banghart, J. (November 2009). NIST Special Publication 800-126 The Technical Specification for the Security Content Automation Protocol (SCAP): SCAP Version 1.0 Recommendations of the National Institute of Standards and Technology. Retrieved on July 6, 2014, from http://csrc.nist.gov/publications/nistpubs/800-126/sp800-126.pdf

*[3] Radack, S. (ed.). (September 2010). ITL BULLETIN FOR SEPTEMBER 2010. SECURITY CONTENT AUTOMATION PROTOCOL (SCAP): HELPING ORGANIZATIONS MAINTAIN AND VERIFY THE SECURITY OF THEIR INFORMATION SYSTEMS. Retrieved on July 6, 2014, from http://csrc.nist.gov/publications/nistbul/september2010-bulletin.pdf

[4] National Institute of Standards and Technology. (January 24, 2014). SCAP Specifications. Security Content Automation Protocol. Retrieved on July 6, 2014, from http://scap.nist.gov/revision/1.0/index.html

[5] National Institute of Standards and Technology. (n.d.). National Vulnerability Database Version 2.2. National Vulnerability Database. Retrieved on July 6, 2014, from http://nvd.nist.gov/home.cfm

[6] The MITRE Corporation. (July 9, 2014). Open Vulnerability and Assessment Language. OVAL. Retrieved on July 10, 2014, from http://oval.mitre.org/

[7] The MITRE Corporation. (June 11, 2014). Common Vulnerabilities and Exposures. CVE. Retrieved on July 10, 2014, from http://cve.mitre.org/

[8] National Institute of Standards and Technology. (July 20, 2014). Common Configuration Enumeration (CCE) Reference Data. National Vulnerability Database. Retrieved on July 20, 2014, from http://nvd.nist.gov/cce

[9] National Institute of Standards and Technology. (July 20, 2014). Official Common Platform Enumeration (CPE) Dictionary. National Vulnerability Database. Retrieved on July 20, 2014, from http://nvd.nist.gov/cpe.cfm

[10] National Institute of Standards and Technology. (January 24, 2014). XCCDF - The Extensible Configuration Checklist Description Format. Security Content Automation Protocol. Retrieved on July 10, 2014, from http://scap.nist.gov/specifications/xccdf/

[11] Defense Information Systems Agency. (July 9, 2014). Security Content Automation Protocol (SCAP) Content and Tools. SCAP STIG Automated Content. Retrieved on July 10, 2014, from http://iase.disa.mil/stigs/scap/index.html

[12] Wikipedia. (April 4, 2014). Security Content Automation Protocol. Wikipedia, The Free Encyclopedia. Retrieved on July 15, 2014, from http://en.wikipedia.org/wiki/Security_Content_Automation_Protocol

[13] National Institute of Standards and Technology. (July 20, 2014). Security Content Automation Protocol Validated Products. National Vulnerability Database. Retrieved on July 20, 2014, from http://nvd.nist.gov/scapproducts.cfm

[14] National Institute of Standards and Technology. (January 24, 2014). SCAP Related Publications. Security Content Automation Protocol. Retrieved on July 15, 2014, from http://scap.nist.gov/publications/

[15] The MITRE Corporation. (July 9, 2014). About OVAL. Open Vulnerability and Assessment Language. OVAL. Retrieved on July 15, 2014, from http://oval.mitre.org/about/index.html

*[16] Quinn, S., Scarfone, K., Barrett, M., Johnson, C. (July 2010). NIST Special Publication 800-117 Guide to Adopting and Using the Security Content Automation Protocol (SCAP) Version 1.0. Recommendations of the National Institute of Standards and Technology. Retrieved on July 14, 2014, from http://csrc.nist.gov/publications/nistpubs/800-117/sp800-117.pdf

[17] National Institute of Standards and Technology. (January 24, 2014). Security Content Automation Protocol (SCAP) Validation Program. Security Content Automation Protocol. Retrieved on July 16, 2014, from http://scap.nist.gov/validation/index.html

[18] National Institute of Standards and Technology. (January 24, 2014). SCAP Specifications. Security Content Automation Protocol. Retrieved on July 17, 2014, from http://scap.nist.gov/revision/index.html

*[19] Banghart, J., Cook, M., Quinn, S., Waltermire, D., Bove, A. (January 2013). NISTIR 7511 Revision 3 Security Content Automation Protocol (SCAP) Version 1.2 Validation Program Test Requirements. Retrieved on July 17, 2014, from http://nvlpubs.nist.gov/nistpubs/ir/2013/NIST.IR.7511.pdf

Key Terms

ACS – Authenticated Configuration Scanner

CCE – Common Configuration Enumeration

CPE – Common Platform Enumeration

CVE – Common Vulnerabilities and Exposures

CVSS – Common Vulnerability Scoring System

DHS – Department of Homeland Security

DISA – Defense Information Systems Agency

DoD – Department of Defense

FDCC - Federal Desktop Core Configuration

FISMA - Federal Information Security Management Act

FSO – DISA Field Security Operations

HIPAA – Health Insurance Portability and Accountability Act

ITL – Information Technology Library

ISO – International Organization for Standardization

NIST – National Institute of Standards and Technology

NVD – National Vulnerability Database

NVLAP - National Voluntary Laboratory Accreditation Program

OVAL – Open Vulnerability and Assessment Language

SOX – Sarbanes-Oxley

SP – Special Publication

USGCB – United States Government Configuration Baseline

XCCDF - Extensible Configuration Checklist Description Format

XML – Extensible Markup Language