A concept of standard-based vulnerability management automation for IT systems

R. KASPRZYK, A. STACHURSKI
rkasprzyk@wat.edu.pl

Institute of Computer and Information Systems
Faculty of Cybernetics, Military University of Technology
Kaliskiego Str. 2, 00-908 Warsaw

The paper focuses on the attempt to show a way of automating IT vulnerability management across enterprise systems with the use of the Security Content Automation Protocol. SCAP offers a set of components which provide, among others, adjustable security checklists, standardised dictionaries of security vulnerabilities and vulnerability scoring methods that may prove valuable for organisations in terms of security analysis activities and quantitative risk assessment.

Keywords: vulnerabilities, SCAP, security.

1. Introduction

Together with advances in information technology and spread of Internet services, increased criminal activity has become present in today's cyberspace. Hence, cyber security heavily depends on technical capabilities to accurately respond to various kinds of threats to cyberspace integrity and information safety. Last decades have shown a large number of incidents that prove how important effective security management is. For this purpose a variety of spectacular cyber-attacks, can be mentioned:

- Cyber-attacks on government IT infrastructure in ESTONIA in 2007 [1];
- Cyber-attacks conducted by both sides of the armed conflict in South Ossetia in 2008 [2];
- Cyber-attacks on Polish government infrastructure after implementing ACTA (*Anti-Counterfeiting Trade Agreement*) [3];
- Cyber-attacks aimed at sabotaging Iran's nuclear program with the use of advanced malicious software: *Stuxnet* (2010) [4], *Duqu* (2011) [5], *Flame* (2012) [6] and many more.

To provide appropriate information safety measures it is necessary to constantly update the knowledge of cyber threats, perform their complex analyses, as well as implement new methods for minimizing potential vulnerabilities impact in the future. Therefore, it is strongly advisable to acquire appropriate tools that would allow for automation of dedicated IT security management processes.

The article presents a description of a concept of vulnerability management automation in IT systems. The key assumption of the proposed solution includes the use of SCAP (*The Security Content Automation Protocol*) components which are described in the following sections.

2. Overview of cyber vulnerabilities

These days no system can be considered as 100% secure; thus each one is exposed to a variety of vulnerabilities. In terms of security of information systems, vulnerability can be generally understood as a weakness of a system or its component, which allows an attacker to compromise its confidentiality, integrity and availability [7]. When exploited, vulnerabilities can be used for many purposes, usually causing unexpected and harmful effects on the victim. Therefore, organisations responsible for cyber security draw attention to the importance of improving the knowledge and updating methods of vulnerability prevention.

Among many classifications of vulnerabilities, American institute NIST, divided them into three basic groups as follows:

- Software flaw vulnerabilities – caused by unintentional errors made in the process of designing and coding of software [7];
- Security configuration issue vulnerabilities – faulty elements of software security configuration which can be accessed and modified through the software itself [8];
- Software feature misuse vulnerabilities – associated with the inappropriate use of the

software feature which may lead to compromise the security of the system. Misuse cases in general include additional features of software for an attacker (e.g. transferring malicious files via an email or sending malicious links) [9].

As there may be lots of vulnerabilities on every system, these may have a wide range of characteristics. Some tend to be easily exploitable, while others may require from an attacker more efforts or the use of more sophisticated techniques like e.g. social engineering. As long as vulnerability characteristics can be measured and documented in a consistent way, organisations will then be able to decide which vulnerabilities they will have to focus on with priority.

3. Structured representation of security-related data

When planning and creating a system organizations often produce their unique security solutions. Nevertheless, these should be both machine and human-readable to be effectively managed and verified in an automated manner. One of the few approaches to manage security in IT systems, which can be considered as supposedly universal, is implementing SCAP.
SCAP (*The Security Content Automation Protocol*) is a method for implementing standards to automate the process of:

- Vulnerability management;
- Security measurement;
- policy compliance evaluation in IT systems [10] [11].

SCAP in version 1.2, released in September 2011, comprises 11 individual standards (referred to as components) [12]. These are used to provide a formal and unified description of various aspects of IT system security and offer methods to seek and score vulnerabilities to measure their possible impact. SCAP components are divided into following specifications [13]:

a) Languages – allow for formal representation of security policies, mechanisms, security tests and their results:
   - OVAL 5.1 (*Open Vulnerability and Assessment Language*);
   - XCCDF 1.2 (*Extensible Configuration Checklist Description Format*);
   - OCIL 2.0 (*Open Checklist Interactive Language*).

b) Reporting Formats – enable formalized expression of information processed by the systems that manage security processes:
   - ARF 1.1 (*Asset Reporting Format*);
   - AI 1.1 (*Asset Identification*).

c) Enumerations – provide a standard naming format and dictionaries of artefacts related to IT security:
   - CPE 2.3 (*Common Platform Enumeration*);
   - CCE 5 (*Common Configuration Enumeration*);
   - CVE (*Common Vulnerabilities and Exposures*).

d) Measurement and scoring systems – methods for evaluating the characteristics of individual vulnerabilities and configuration weaknesses:
   - CVSS 2.0 (*Common Vulnerability Scoring System*);
   - CCSS 1.0 (*Common Configuration Scoring System*).

e) Integrity – describes mechanisms to protect data integrity of previously mentioned components:
   - TMSAD 1.0 (*Trust Model for Security Automation Data*).

SCAP defines how components listed above are combined together, however, it is not a mandatory requirement to use all of them. For some organisations it is sufficient in terms of defining security policy to select most common SCAP components, as depicted in Figure 1.

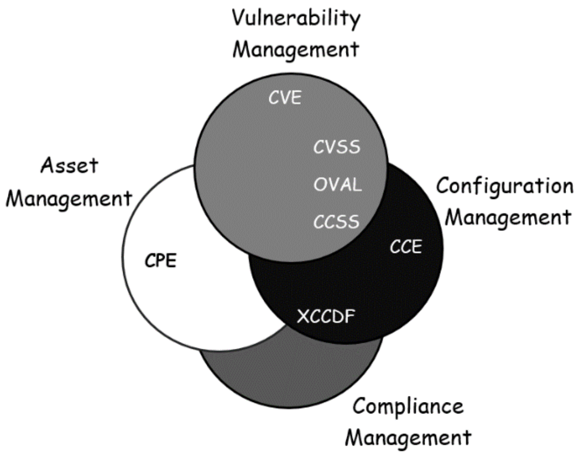


Fig. 1. Key SCAP Components

Common uses and more detailed specifications of key SCAP components are described in the following sections.

4. Standardized enumerations

The vast majority of IT organizations use diverse software for their security management. This includes vulnerability scanners or intrusion detection systems which use their independent data formats, interfaces and naming conventions. Such a policy limits the interoperability and exchanging security data with external organizations requires a variety of manual customizations. To unify this approach, SCAP provides a standard nomenclature and official dictionaries for three sets of specified identifiers (referred as to enumerators) as follows:

- CVE (*Common Vulnerabilities and Exposures*) is a dictionary of publicly known information security vulnerabilities and exposures providing common names for officially unveiled cyber security issues [14].
- CCE (*Common Configuration Enumeration*) is a dictionary of unique identifiers referring to known security system configuration issues and configuration guidance statements which can be defined as preferred or required settings or policy for a computer system [15].
- CPE (*Common Platform Enumeration*) is a standardized method that allows for identifying and specifying classes of applications, operating systems, and hardware components available among enterprise's assets [16].

Using CVE, CCE, and CPE may simplify the process of information sharing across a wide spectrum of systems, repositories and services. It also provides general guidance on detected security issues. For instance, departments from one organisation which use different SCAP – validated tools for their internal security scans can automatically combine their outputs into one consistent report for the whole organisation. Standardized enumerations are fundamental elements of security checklists described in the next section.

5. Security checklists

The general purpose of implementing SCAP security checklists is to automate the process of security policy verification, testing and configuration assessment. Checklists, based on CVE, CCE and CPE dictionaries, are sets of rules which include configuration settings, software versions, patch levels and many other important characteristics in terms of system security. These can be interpreted by dedicated SCAP content consumers and compared with the settings of the examined system to indicate possible deviations from the particular security requirements imposed by an organization. The structure of each checklist is expressed using the following SCAP components:

- XCCDF (*The Extensible Configuration Checklist Description Format*);
- OVAL (*Open Vulnerability and Assessment Language*);
- OCIL (*The Open Checklist Interactive Language*).

XCCDF is an XML-based language for defining technical and non-technical security checklists, benchmarks and ways recording assessment results in a standardized format [17]. XCCDF document includes references to OVAL and OCIL definitions as shown in Figure 2.

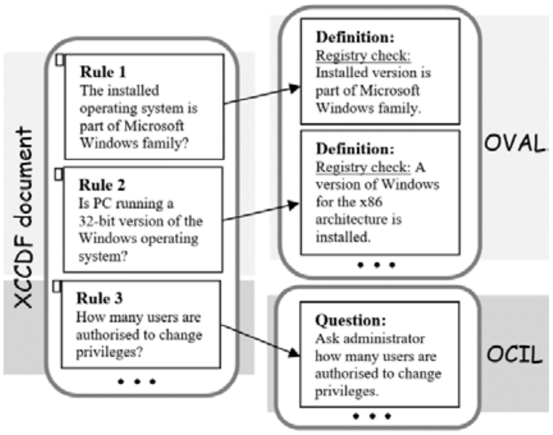

XCCDF document
Rule 1
The installed operating system is part of Microsoft Windows family?
Rule 2
Is PC running a 32-bit version of the Windows operating system?
Rule 3
How many users are authorised to change privileges?
Definition:
Registry check:
Installed version is part of Microsoft Windows family.
Definition:
Registry check: A version of Windows for the x86 architecture is installed.
OVAL
Question:
Ask administrator how many users are authorised to change privileges.
OCIL

Fig. 2. XCCDF – sample use case scenario

OVAL is a standard that promotes publicly available security content and standardizes its flow across a variety of tools and services [18]. It includes an XML-based language that defines the way of: representing characteristics of the system, checking the presence of specified machine state (vulnerable, compliant, installed application, patch state, etc.) and reporting results of OVAL tests.

Each OVAL definition includes a set of tests and criteria. Criteria are logical operators which indicate how to produce a final result of tests within the definition. Tests include parameters which are references to particular objects in the system to be examined (e.g. registry paths, file paths) and their required states as shown in Figure 3.

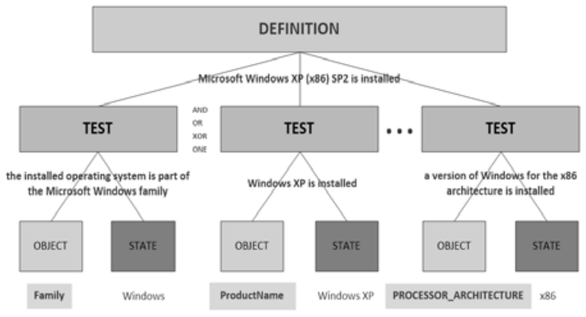
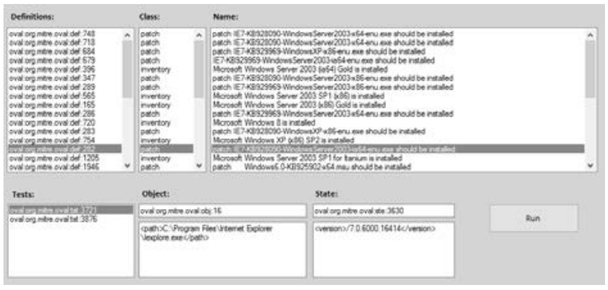

DEFINITION
Microsoft Windows XP (x86) SP2 is installed
TEST
AND
OR
XOR
ONE
TEST
TEST
the installed operating system is part of the Microsoft Windows family
Windows XP is installed
a version of Windows for the x86 architecture is installed
OBJECT
STATE
OBJECT
STATE
OBJECT
STATE
Family
Windows
ProductName
Windows XP
PROCESSOR_ARCHITECTURE
x86

Fig. 3. A structure of a sample OVAL definition

Conducting OVAL tests requires the use of dedicated software that allows for interpreting OVAL documents. In some cases OVAL interpreters list input definitions and divide their elements into separate groups. It depicts the structure of an input test file and as a result provides a convenient way of viewing definitions, their particular tests, metadata and references to examined objects. The example of such an implementation was introduced in the author's software and is presented in Figure 4.



Definitions:	Class:	Name:
oval:org.mitre.oval:def:748	patch	patch IE7-KB928090-WindowsServer2003-x64-enu.exe should be installed
oval:org.mitre.oval:def:713	patch	patch IE7-KB928090-WindowsServer2003-x64-enu.exe should be installed
oval:org.mitre.oval:def:684	patch	patch IE7-KB929969-WindowsXP-x86-enu.exe should be installed
oval:org.mitre.oval:def:679	patch	IE7-KB929969-WindowsServer2003-ia64-enu.exe should be installed
oval:org.mitre.oval:def:396	inventory	Microsoft Windows Server 2003 (ia64) Gold is installed
oval:org.mitre.oval:def:347	patch	patch IE7-KB928090-WindowsServer2003-x86-enu.exe should be installed
oval:org.mitre.oval:def:289	patch	patch IE7-KB929969-WindowsServer2003-x86-enu.exe should be installed
oval:org.mitre.oval:def:565	inventory	Microsoft Windows Server 2003 SP1 (x86) is installed
oval:org.mitre.oval:def:165	inventory	Microsoft Windows Server 2003 (x86) Gold is installed
oval:org.mitre.oval:def:286	patch	patch IE7-KB929969-WindowsServer2003-x64-enu.exe should be installed
oval:org.mitre.oval:def:720	inventory	Microsoft Windows 8 is installed
oval:org.mitre.oval:def:283	patch	patch IE7-KB928090-WindowsXP-x86-enu.exe should be installed
oval:org.mitre.oval:def:754	inventory	Microsoft Windows XP (x86) SP2 is installed
oval:org.mitre.oval:def:282	patch	patch IE7-KB928090-WindowsServer2003-ia64-enu.exe should be installed
oval:org.mitre.oval:def:1205	inventory	Microsoft Windows Server 2003 SP1 for Itanium is installed
oval:org.mitre.oval:def:1946	patch	patch Windows6.0-KB925902-x64.msu should be installed

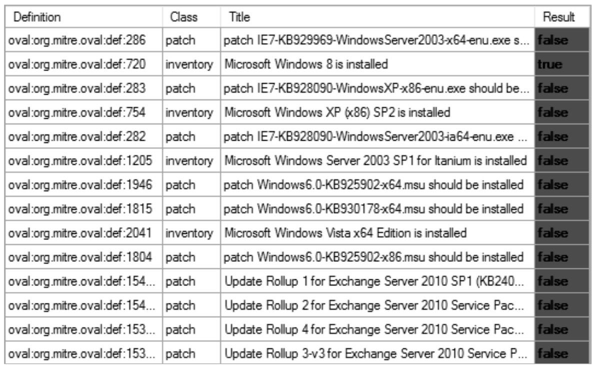
Tests:
oval:org.mitre.oval:tst:[illegible]
oval:org.mitre.oval:tst:3876

Object:
oval:org.mitre.oval:obj:16
<path>C:\Program Files\Internet Explorer \iexplore.exe</path>

State:
oval:org.mitre.oval:ste:3630
<version>7.0.6000.16414</version>

Fig. 4. OVAL structure divided into groups

In the next stage testing is conducted on the basis of input data from OVAL definitions. Then corresponding test results are displayed, similarly to the style used in author's application presented in Figure 5.



Definition	Class	Title	Result
oval:org.mitre.oval:def:286	patch	patch IE7-KB929969-WindowsServer2003-x64-enu.exe s...	false
oval:org.mitre.oval:def:720	inventory	Microsoft Windows 8 is installed	true
oval:org.mitre.oval:def:283	patch	patch IE7-KB928090-WindowsXP-x86-enu.exe should be...	false
oval:org.mitre.oval:def:754	inventory	Microsoft Windows XP (x86) SP2 is installed	false
oval:org.mitre.oval:def:282	patch	patch IE7-KB928090-WindowsServer2003-ia64-enu.exe ...	false
oval:org.mitre.oval:def:1205	inventory	Microsoft Windows Server 2003 SP1 for Itanium is installed	false
oval:org.mitre.oval:def:1946	patch	patch Windows6.0-KB925902-x64.msu should be installed	false
oval:org.mitre.oval:def:1815	patch	patch Windows6.0-KB930178-x64.msu should be installed	false
oval:org.mitre.oval:def:2041	inventory	Microsoft Windows Vista x64 Edition is installed	false
oval:org.mitre.oval:def:1804	patch	patch Windows6.0-KB925902-x86.msu should be installed	false
oval:org.mitre.oval:def:154...	patch	Update Rollup 1 for Exchange Server 2010 SP1 (KB240...	false
oval:org.mitre.oval:def:154...	patch	Update Rollup 2 for Exchange Server 2010 Service Pac...	false
oval:org.mitre.oval:def:153...	patch	Update Rollup 4 for Exchange Server 2010 Service Pac...	false
oval:org.mitre.oval:def:153...	patch	Update Rollup 3-v3 for Exchange Server 2010 Service P...	false

Fig. 5. An example of OVAL test results grid

Such an approach allows for generating reports used to show the state of the examined system.

OCIL defines a framework for expressing questionnaires that can be used by software to harvest information stored during previous data collection efforts or to collect information from people [19]. Unlike in OVAL use case scenario, OCIL is used to manually verify the level of security, on the basis of an interview with the human. For this purpose OCIL provides standardized forms of questions, instructions, answer sheets and any necessary guidance documents. An example of OCIL test is questioning an administrator about an IT infrastructure within organization (e.g. "How many servers do you have?", "What types of security passes do employees have? ", etc.).

XCCDF, OVAL and OCIL used to create security checklists are complementary for each other. Equally, these make security practices for dedicated systems consistent and when used regularly, allow for continuous monitoring of their security.

6. Security measurement

Organizations which use methods of quantitative measurement and scoring vulnerabilities can derive considerable benefits. This may include the ability to predict how cyber-attackers can compromise the security of their systems, minimize the risk or evaluate the potential outcomes of being attacked. For this purpose in combination with CVE, CCE and CPE, two SCAP components can be used as follows:

- CVSS – a standard for measuring the impact of software flaw vulnerabilities and a format for communicating vulnerability characteristics [7];
- CCSS – a standard for measuring the impact of security configuration issue vulnerabilities based on CVSS [8].

As a complement to CVSS and CCSS American institute NIST created CMSS (*Common Misuse Scoring System*), which is a concept designed for measuring and scoring of software feature misuse vulnerabilities. These group of vulnerabilities allow attackers to use software functionalities for malicious purposes [9].

The idea of using mentioned scoring methods is relatively similar. For each of them three groups of metrics can be distinguished: Base Metrics, Temporal Metrics and Environmental Metrics, defined by equations. Base Metrics describe the characteristics of the examined vulnerability that are constant over time and across user environments. Temporal metrics refer to vulnerabilities which can change over time while environmental metrics customize both base and temporal scores on characteristics of a particular user environment [8]. Both environmental and temporal metrics are optional and provide additional measurement parameters, which produce more accurate scoring results of the vulnerability.

The vulnerability itself is expressed as a vector represented by a structured string of characters, which are related to specific vulnerability characteristics. To generate a score, individual vector components are extracted and converted into corresponding numeric values. Then they are substituted in the equation formulas and a result between 0 and 10 is returned – see Figure 6.

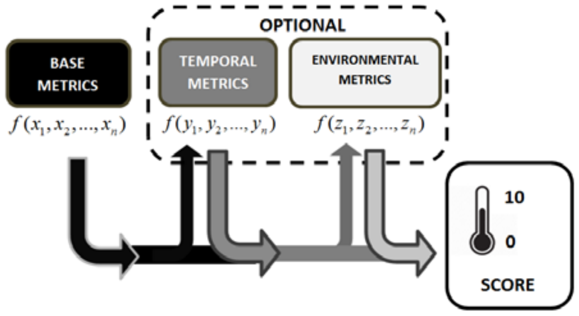
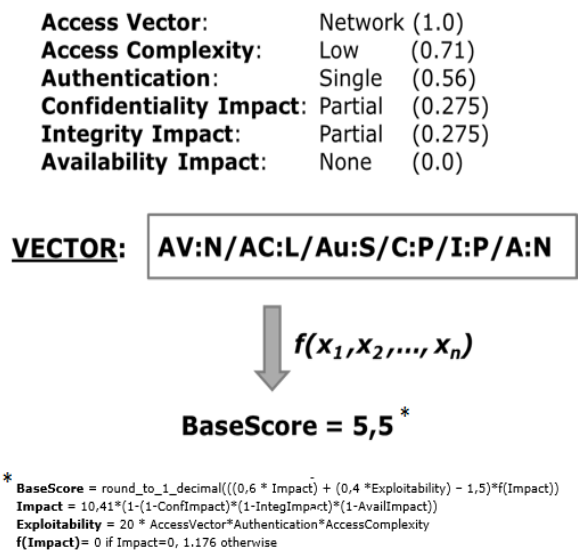

OPTIONAL
BASE METRICS
TEMPORAL METRICS
ENVIRONMENTAL METRICS
$f(x_1, x_2, \dots, x_n)$
$f(y_1, y_2, \dots, y_n)$
$f(z_1, z_2, \dots, z_n)$
10
0
SCORE

Fig. 6. A scheme of vulnerability scoring process

In this case, the lower result returned, the lower vulnerability severity. A scheme of calculating sample Base Metrics in CVSS 2.0 is presented in Figure 7. The sample scoring SQL Injection for MySQL Database based on paper [7].



Access Vector:	Network	(1.0)
Access Complexity:	Low	(0.71)
Authentication:	Single	(0.56)
Confidentiality Impact:	Partial	(0.275)
Integrity Impact:	Partial	(0.275)
Availability Impact:	None	(0.0)

VECTOR: **AV:N/AC:L/Au:S/C:P/I:P/A:N**

$f(x_1,x_2,\dots,x_n)$

BaseScore = 5,5 *

* **BaseScore** = round_to_1_decimal(((0,6 * Impact) + (0,4 *Exploitability) – 1,5)*f(Impact))
Impact = 10,41*(1-(1-ConfImpact)*(1-IntegImpact)*(1-AvailImpact))
Exploitability = 20 * AccessVector*Authentication*AccessComplexity
f(Impact)= 0 if Impact=0, 1.176 otherwise

Fig. **7**. A sample CVSS 2.0 BaseScore calculation

The equation formulas $f(x_1, x_2, \dots ,x_n)$ for calculating CVSS scores are specified by experts such as vulnerability bulletin analysts, application and security product vendors who own the most detailed information about vulnerabilities and their impact on cybersecurity. Vulnerability scoring carried out by dedicated software may consequently prompt the organisations which vulnerabilities have to be addressed with priority in their solutions.

7. Summary

Since one of the basic assumptions of defining requirements for IT systems is to provide their effective security management, it is necessary to develop an appropriate security policy. To improve this process, American Institute NIST created SCAP that is a standardized and automated approach to managing security of IT systems. SCAP is widely promoted by US government and has also been implemented in a large number of commercial worldwide products that support information security. This article presents a concept of vulnerability management automation for IT systems which is based on selected SCAP components. Each SCAP component focuses on specific areas related to security issues and provides a standardized format for documenting system security settings and configuration mechanisms.

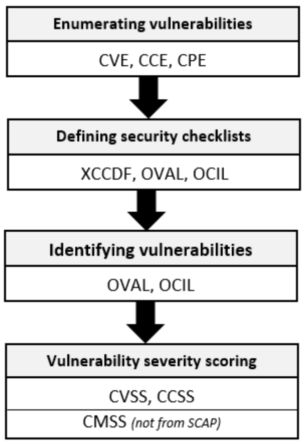

Enumerating vulnerabilities
CVE, CCE, CPE
Defining security checklists
XCCDF, OVAL, OCIL
Identifying vulnerabilities
OVAL, OCIL
Vulnerability severity scoring
CVSS, CCSS
CMSS (not from SCAP)

Fig. 8. A simplified view of the proposed concept

These include gathering information about vulnerabilities, monitoring and scanning systems for publicly known vulnerabilities and configuration issues, checking systems for signs of compromise, scoring vulnerabilities and some more. Many use scenarios may depend on communication with other applications, e.g. antivirus program, firewall or tools for performing penetration tests. The general purpose of SCAP components implementation is then making security more or less measurable, which eventually may lead to decreasing the risk of potential cyber threats.

8. Bibliography

[1] Kari A., "An Exceptional war That Ended in Victory for Estonia or an Ordinary e-Disturbance? Estonian Narratives of the Cyber-Attacks in 2007", *European Conference on Information Warfare and Security*, July 2012.

[2] Hollis D., "Cyberwar Case Study: Georgia 2008", Small Wars Journals, http://smallwarsjournal.com/blog/journal/docs-temp/639-hollis.pdf.

[3] CERT, "DDoS against Polish government websites", 23 January 2012. [Online]. http://www.cert.pl/news/4856/langswitch_lang/en.

[4] "Thanks, Stuxnet", *New Scientist*, Vol. 210, 2 April 2011.

[5] "Duqu: son of Stuxnet?", *Computer Fraud & Security*, Vol. 2011(11), 3 (2011).

[6] Higgins K.J., "Flame Gives Spyware A Next-Gen Updat", *Information Week*, 11 June 2012.

[7] Mell P., Scarfone K. and Romanosky S., "A complete guide to the common vulnerability scoring system version 2.0", NIST, 2007.

[8] Scarfone K. and Mell P., "The common configuration scoring system (ccss): Metrics for software security configuration vulnerabilities", NIST, 2010.

[9] LeMay E., Scarfone K. and Mell P., "The Common Misuse Scoring System (CMSS): Metrics for Software Misuse Vulnerabilities", NIST, 2012.

[10] NIST, "The Security Content Automation Protocol (SCAP)", [Online]. http://scap.nist.gov.

[11] Wikipedia, "Security Content Automation Protocol", [Online]. https://en.wikipedia.org/wiki/Security_Content_Automation_Protocol.

[12] NIST, "Security Content Automation Protocol (SCAP) Specifications", [Online]. http://scap.nist.gov/revision/1.2/index.html.

[13] Waltermire D., Quinn S., Scarfone K. and Halbardier A., "The Technical Specification for the Security Content Automation Protocol (SCAP): SCAP Version 1.2", NIST, 2011.

[14] MITRE, "Common Vulnerabilities and Exposures. The Standard for Information Security Vulnerability Names" [Online] https://cve.mitre.org.

[15] MITRE, "Common Configuration Enumeration – Standardized Identifiers for Security Configuration Issues and Exposures", 2012.

[16] NIST, "Common Platform Enumeration (CPE)", http://scap.nist.gov/specifications/cpe/.

[17] Waltermire D., Schmidt C., Scarfone C., and Ziring N., "Specification for the Extensible Configuration Checklist Description Format (XCCDF) Version 1.2", 2011.

[18] MITRE, "An Introduction to the OVAL™ Language", MITRE, 2006.

[19] Waltermire D., Scarfone C. and Casipa M., "Specification for the Open Checklist Interactive Language (OCIL) Version 2.0", 2011.

Oparta na standardach koncepcja zarządzania podatnościami systemów teleinformatycznych na zagrożenia

R. KASPRZYK, A. STACHURSKI

Celem artykułu jest przedstawienie próby automatyzacji zarządzania podatnościami systemów teleinformatycznych przy zastosowaniu grupy standardów wchodzącej w skład SCAP (*The Security Content Automation Protocol*). Cel ten może zostać osiągnięty między innymi poprzez zdefiniowanie standardowych formatów nazw i słowników artefaktów związanych z bezpieczeństwem systemów teleinformatycznych, tworzenie kwestionariuszy pozwalających zarówno na manualną i programową ewaluację zgodności z założoną polityką bezpieczeństwa, jak i na badania charakterystyk konkretnych podatności. Działania te mogą wspomóc czynności związane ze szczegółową analizą bezpieczeństwa systemów IT, jak również z szacowaniem ryzyka potencjalnego ataku cybernetycznego.

Słowa kluczowe: podatność na cyberzagrożenia, SCAP, bezpieczeństwo teleinformatyczne.