Analytical Attack Modeling and Security Assessment based on the Common Vulnerability Scoring System

Elena Doynikova [1], Andrey Chechulin [1], Igor Kotenko [1,2]
[1]St. Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences (SPIIRAS), St. Petersburg, Russia
{doynikova, chechulin, ivkote}@comsec.spb.ru
[2] St. Petersburg National Research University of Information Technologies, Mechanics and Optics, St. Petersburg, Russia

***Abstract*—The paper analyzes an approach to the analytical attack modeling and security assessment on the base of the Common Vulnerability Scoring System (CVSS) format, considering different modifications that appeared in the new version of the CVSS specification. The common approach to the analytical attack modeling and security assessment was suggested by the authors earlier. The paper outlines disadvantages of previous CVSS version that influenced negatively on the results of the attack modeling and security assessment. Differences between new and previous CVSS versions are analyzed. Modifications of the approach to the analytical attack modeling and security assessment that follow from the CVSS modifications are suggested. Advantages of the modified approach are described. Case study that illustrates enhanced approach is provided.**

I. Introduction

The problem of monitoring the security of computer networks is important and relevant, especially in modern conditions, when the activities of a growing number of organizations depend on their secure and reliable operation.

Modern hackers to achieve their goals often implement complex multi-stage attacks that involve sequence of steps, based on the exploitation of various vulnerabilities, configuration errors and peculiarities of implementation of software and hardware. Timely detection of attacker in the system and accurate prediction of its objectives can help prevent serious damages to the system and to avoid large losses. For these purposes, researchers have developed approaches based on analytical modeling.

Many methods for modeling the attacker steps in the system were proposed, including those in the form of graphs of attack actions [1-12]. The authors of this paper proposed their own approach earlier [13-16]. An important feature of the authors' approach is use of open bases, which on the one hand allows to take into account the maximum number of known vulnerabilities, and on the other hand helps to automate the process. Another important feature of the previously proposed approach is the efficiency of the graph construction, which is critical in the dynamic mode of the system, when the level of overall loss depends on timely respond to an attack.

The Common Vulnerability Scoring System (CVSS) is in the basis of the proposed approach [17]. The key factors due to which this format was chosen is the openness of CVSS ratings, which allows to use them to produce own security indexes, and existence of the links between CVSS and the Common Platform Enumeration (CPE) [18] and the Common Configuration Enumeration (CCE) [19], which allows to automate the identification and assessment of vulnerabilities.

The previous format of CVSS had several features that required a number of assumptions at automatic generation of attack graphs. In 2015 the new version of CVSS was published, which took into account the problems of the previous version.

In this study we consider the new format of CVSS, analyze its advantages and its impact on the approach proposed by us earlier. On the basis of the performed analysis we propose a new method of the graph generation and the security analysis which uses the CVSS of version 3.0. We consider the advantages and disadvantages of the new method by example. The paper is essentially the first attempt of such an analysis and demonstrates the possibility of using the proposed approach with the application of the CVSS of version 3.

Thus the main contribution of this paper is analysis of the CVSS of version 3.0, the comparison of the CVSS of version 3.0 with CVSS of version 2.0, the new method of attack modeling and security analysis on the base of the CVSS of version 3.0, analysis of advantages and disadvantages of the new method. The common algorithm of the new method contains the same stages as the previous one [13-16]. The differences are the CVSS indexes that are used on different stages of the algorithm, and equations that are used for the security assessment.

The paper is structured as follows. Section II reviews main related works in the area of the CVSS and analytical attack modeling. Section III describes CVSS of version 2.0 and version 3.0, and analyses their differences. Section IV introduces our previous and new approach to the analytical attack modeling and security analysis. Case study and discussion are provided in the Section V. Finally, main results of the research and future work are presented in conclusion.

II. Related work

The issues of formation of attack graphs and security analysis based on them are considered in many works [1-12].

There are several types of attack graphs outlined:

(1) complete graph of attacks [1] – includes all the ways an attacker can compromise the network;

(2) the predictive graph [2], where the node is added to the graph if no ancestor of this node uses the same vulnerability for moving to the same condition as the new node;

(3) graph with many preconditions [3] – includes three types of nodes (condition; precondition; vulnerability), and additional circular arcs to show the relationships with the already existing nodes.

Several papers considered the problem of operativeness in the construction of attack graphs [3], [4], [16].

Basing on attack graphs there were developed several probabilistic models for the analysis of system security. Probabilistic attack graphs are suggested to use in [4-7]. In [8-12] the Bayesian attack graphs are applied.

To assess the security different vulnerability scoring systems can be used, including systems that are based on the qualitative ranking (SANS Institute's Critical Vulnerability Analysis Scale, Microsoft Security Bulletin Severity Rating System), systems that are based on the quantitative ranking (PCI DSS) and systems that are based on the integrated metrics (CVSS [20] and nCircle vulnerability scoring system).

The authors approach to the attack modeling and security assessment is based on the CVSS. This system was selected because of the following aspects: the openness of CVSS ratings, which allows to use them to produce own security indexes and model the pre and post conditions of the vulnerabilities exploitation, the availability of the CVSS scores in the open databases of vulnerabilities, the reliability of the CVSS scores because they are defined by the group of security experts, the links between CVSS and Common Platform Enumeration (CPE) [18] and Common Configuration Enumeration (CCE) [19], which allows to automate the identification and assessment of vulnerabilities.

In previous papers of the authors there were presented different versions of the algorithms for constructing and analyzing attack trees [13], [14], metrics calculated basing on attack trees [15] and modification of algorithms for constructing attack trees to generate and analyze models of attacks in near real time [16].

However, the approach proposed in these studies had some limitations related to the restrictions of the CVSS format. In this paper, we will examine the format changes introduced in the new version, and analyze their influence on the previously proposed approach.

III. CVSS AS THE BASIS FOR THE ATTACK TREE GENERATION

A. CVSS of version 2.0

CVSS includes a number of indexes that characterize the vulnerabilities of hardware and software that allow to obtain a final integrated vulnerability assessment that defines its severity compared to other vulnerabilities [20]. CVSS consists of three groups of indexes: basic, temporal, and contextual. At the moment to construct the graph they use only basic indexes, so they are briefly described below. Values for known vulnerabilities can be found in open vulnerability database NVD [21].

Group of base indexes of the CVSS of version 2.0 includes two groups of indexes: (1) Exploitability (which define the method of access to the vulnerability, and whether additional conditions for its operation are needed) and (2) Impact (which depicts how the vulnerability will affect an asset in the case of exploitation).

The indexes of the group Exploitability include:

- *Access Vector (AV)* – determines how the vulnerability is exploited (the more remote intruder may attack the host, the higher is the vulnerability assessment). If the vulnerability can be exploited in several ways, then only the most remote access is selected. When forming the graph, this index is used to define the preconditions of the exploitation that allows to create serial communication between them, to combine steps of the attack into a multi-step attack. *Uncertainty* 1 represents the value of the index "local access" to note if the physical or logical access to a computer is used.
- *Access Complexity (AC)* – defines the complexity of the attack to be undertaken for the exploitation of the vulnerability after the penetrator has gained access to the system. The lower the complexity is, the higher the vulnerability assessment is. This index allows you to determine how likely a successful exploitation of the vulnerability is. *Uncertainty* 2 shows that vulnerabilities that require additional action from the user are not considered separately.
- *Authentication* (Au) – specifies how many times an attacker must authenticate to the system to exploit the vulnerability (the complexity of the process is not considered, only the number is). The less is the identity, the higher is the value. This index differs from the Access Vector, i.e. it is considered that the access to the system is already obtained (additionally to the login you need to provide additional authentication). *Uncertainty* 3 represents the privileges level under additional authentication.

The index of the group Impact include:

- *Confidentiality Impact* (C) – determines the damage for your privacy as the result of successful exploitation. Increase of confidentiality damage increases vulnerability assessment.
- *Integrity Impact* (I) – determines the damage to integrity after successful exploitation. Increase in the damage of integrity leads to increase in vulnerability assessment.
- *Availability Impact* (A) – determines the damage of availability as a result of successful exploitation of the vulnerability. The increase in damage of availability increases the vulnerability assessment.

Indexes in this group determine the postconditions of the exploitation and enable to form links between the vulnerabilities to create multi-step attacks. The *uncertainty* 4 represents the vulnerability scope.

B. CVSS of version 3.0

The important feature of CVSS is the fact that the characterization of vulnerabilities should be obvious to any expert and unambiguous. A number of uncertainties arising from the application of format version 2.0 has been fixed in the new version. CVSS of version 3.0 [22], as well as the previous version, includes two groups of indexes:

(1) Exploitability (Attack Vector, Attack Complexity, Privileges Required, User Interaction) – displays the characteristics of the affected component;

(2) Impact (Confidentiality Impact, Integrity Impact, Availability Impact) – shows the consequences, or affected component.

The most important difference of the new format is that there was additionally added index Scope, which allows to separate the affected component (the component that contains the vulnerability, for example, a software module, driver, etc.) from a component that is damaged (software, hardware, or network resource).

Impact group has not changed compared to the previous version (however, Impact is now defined by Scope, by the maximum impact). In the group Exploitability the index Authentication was changed to Privileges Required. They also added index User Interaction (which was previously taken into account in determining the rating assigned by Access Complexity, and now is separated).

Detailed comparison of indexes of CVSS of version 2.0 and CVSS of version 3.0 and of their values is given in Table I. Lighter colored in the table are indexes /values with small changes, for example, of numerical values, and darker ones are significantly modified indexes/values, for example, newly added or deleted.

TABLE I. COMPARISON OF THE INDEXES OF THE CVSS OF VERSION 2.0 AND CVSS OF VERSION 3.0

CVSS of version 2.0	CVSS of version 3.0
Exploitability group	
Access Vector, AV	**Attack Vector, AV**
AV values	AV values
Local (L): 0.395	Local (L): 0.55 – attacker requires rights to read/write/execute to exploit the vulnerability, that is, the attacker must either be logged, or rely on User Interaction.
	Physical (P): 0.2 – requires physical manipulation of the affected component.
Adjacent Network (A): 0.646	Adjacent (A): 0.62
Network (N): 1.0	Network (N): 0.85
Access Complexity, AC	**Attack Complexity (AC)**
AC values	AC values
High (H): 0.35	High: 0.44 – the success of the attack depends on conditions
Medium (M): 0.61	outside the control of the attacker.
Low (L): 0.71	Low: 0.77 – no special access conditions or extenuating circumstances. The attacker can expect repeated success against the vulnerable component.
Authentication, Au	**Privileges Required, PR**
Au values	PR values
Multiple (M): 0.45	High: 0.27 (0.5 if Scope is Changed) – атакующий авторизован, the attacker is logged in with privileges, giving significant (administrative) access on a vulnerable component that can affect component-wide settings and files.
Single (S): 0.56	Low: 0.62 (0.68 if Scope is Changed) – the attacker is logged in with privileges that provide basic user's capabilities, which affect only the files and settings of the user. Or the attacker can only affect non-confidential resources.
None (N): 0.704	None: 0.85 – the attacker is not authorized, that is, access to settings and files is not required.
–	**User Interaction, UI**
	UI values
	None: 0.85 – the system can be compromised without user intervention.
	Required: 0.62 – the user has to perform some actions before the vulnerability may be exploited. For example, a successful exploit is only possible in case of installation of the application by the system administrator.
Impact group	
Confidentiality Impact, C	**Confidentiality Impact, C**
C values	C values
None (N): 0.0	None (N): 0
Partial (P): 0.275	Low (L): 0.22
Complete (C), 0.660	High (H), 0.56
Integrity Impact, I	**Integrity Impact, I**
I values	I values
None (N): 0.0	None (N): 0
Partial (P): 0.275	Low (L): 0.22
Complete (C): 0.660	High (H): 0.56
Availability Impact, A	**Availability Impact, A**
A values	A values
None (N): 0.0	None (N): 0
Partial (P): 0.275	Low (L): 0.22
Complete (C): 0.660	High (H): 0.56
–	**Scope, S**
	S values
	Unchanged (U) – (the vulnerability only affects resources managed by the same authority: vulnerable and susceptible to the influence component is single
	Changed (C) – the vulnerability affects the resources outside of the privileges of the affected component, in this case vulnerable and susceptible to the influence components are different.

The importance of the index Scope is due to the fact that in version 2.0 it was not clear to what the Impact caused by the vulnerability refers. The index Scope solves this problem by clearly limiting the scope of the impact and thus removing *uncertainty* 4. Scope (S) refers to the set of privileges defined by a computing authority (applications, operating system, sandbox environment), when assigning access to resources (files, CPU, memory, etc.). When the vulnerability of component, managed by one authorization scope, can affect resources managed by other authorization scope, change of Scope occurs. An example would be the vulnerability of a virtual machine (VM) that allows an attacker to delete files on the host OS (maybe even the VM itself). Base estimation is growing in case of change of Scope.

The index Attack Vector (AV) is preserved from the previous version, but its possible values are changed. In version 3.0 the physical access in separated to specific value that eliminates the *Uncertainty* 1 (confusion between local and physical access). The index value the greater, the more remote (logically and physically) attacker can use it (i.e. there are much more remote attackers than those who have physical access to the device).

The index Attack Complexity (AC) is derived from version 2.0 from index Access Complexity, it no longer takes into account interaction with the user, and index values changed (see Table I).

The index Privileges Required (PR), which substituted the index Authentication, is the privilege level required for the attacker before the vulnerability is successfully used. The value of this index takes the maximum value if one does not need any privileges. It removes the *Uncertainty* 3, allowing to link postconditions of the exploitation of the vulnerability on the same host with the preconditions on the other one.

The new index User Interaction (UI) defines the requirements for the user other than the attacker needed for successful compromise of the affected component. The index determines whether a vulnerability may be exploited by attackers desire or an individual user needs to participate (or a process initiated by the user). Index has highest value when the interaction with the user is not required. The introduction of this index removes the *Uncertainty* 2.

Aforementioned indexes are used to calculate the CVSS score for vulnerabilities. For these goals CVSS equations are defined. Due to changes of CVSS indexes and their values, the CVSS equations in version 3.0 are also modified. The CVSS equations of the version 2.0 and version 3.0 are given in the Table II for comparison (designations of the indexes are taken from Table I).

The changes made to CVSS of version 3.0, relieve many of the problems that occurred previously. In the next section we consider in detail the impact of these changes on the process of building and analyzing the attack graph using the proposed approach to analytical attack modeling and security analysis.

IV. Attack trees

A. Generation of Attack Trees based on CVSS 2.0

The general algorithm for constructing attack trees within the frame of the proposed approach to analytical attack modeling and security analysis consists of 3 steps:

(1) formation of matrices by the databases of vulnerabilities and the configuration of software and hardware of hosts;

(2) formation of the lists of the attacks available to attackers;

(3) generation of attack trees based on the connectivity graph of the network and the lists of attack actions.

TABLE II. CVSS equations of the CVSS of version 2.0 and CVSS of version 3.0

CVSS of version 2.0	CVSS of version 3.0
round_to_1_decimal(((0.6* Impact)+(0.4*Exploitability)-1.5)*f(Impact))	If(Impact sub score<=0) CVSS_Score=0, else if Scope Unchanged CVSS_Score=Roundup(min[(Impact +Exploitability), 10]), else if Scope Changed CVSS_Score=Roundup(min[1.08* (Impact+Exploitability), 10])
Impact=10.41*(1-(1-C)*(1-I)*(1-A))	If Scope Unchanged Impact_sub_score = 6.42*ISC_{Base}, else if Scope Changed Impact_sub_score = 7.52*[ISC_{Base}-0.029]-3.25*[ISC_{Base}-0.02][15]
	ISC_{Base}=1-[(1-C)*(1-I)*(1-A)]
Exploitability=20*AV*AC*Au	8.22*AV*AC*PR*UI
if Impact=0 f(Impact)=0, else f(Impact)=1.176	–
round_to_1_decimal – specified to 1 decimal place, that is equal to or higher than its input	Round up – smallest number, specified to 1 decimal place, that is equal to or higher than its input

Let us consider these steps in more detail.

Step 1. For constructing attack trees a list of possible attack actions if formed, with actions separated into groups in accordance with the indexes of CVSS of version 2.0. To do this, for each host of the network the 3-dimensional matrix is built for the following data:

- *class of attacks* (data collection, preparatory actions, elevation of privilege, execution of the target of the attack) is based on the used database (CAPEC [23] or CVE [24]) and the Confidentiality Impact (C), Integrity Impact (I), Availability Impact (A) and Gained Access Level;
- *access type* (remote source without access rights, remote user, local user, administrator) is determined basing on the Access Vector (AV) and Authentication (Au);
- *the level of knowledge of the attacker* (types of vulnerabilities that an intruder will be able to implement) is determined on the basis of Access Complexity (AC).

At that the cells of the matrix (i.e. the intersection of attack class, access type and level of knowledge of the offender) are lists of vulnerabilities, corresponding to these parameters.

After the formation of the matrix itself, its cells are filled with specific attack actions on the basis of the lists of existing vulnerabilities, the respective configurations of software and hardware of the host, and the attacks aimed at collecting information. Lists of possible attacks are limited by the security parameters of the host (with restrictions on the lists of possible vulnerabilities and attacks aimed at collecting information).

As a result, for each host the list of possible attack actions is generated; it is categorized according to the following parameters: class of attack, the required type of access and the required level of knowledge of the intruder. For each group, in its turn, the list of specific attacks and vulnerabilities that these attacks implement is formed. For example, vulnerability CVE-2016-10108 allows to obtain administrator rights (Gained Access Level is “administrator”) on some versions of the Western Digital MyCloud NAS. This vulnerability (or attack action implementing such vulnerability) refers to classes of attacks “privilege elevation” and “attack target execution”, can be exploited remotely (AV value is “Network”) and does not require prior getting of the access rights (AU value is “None”) and the knowledge of the attacker (AC value is “Low”). Thus, the vulnerability belongs to the group of “privilege elevation” (class attack), remote source without access rights (access type) and does not require specialized knowledge (level of knowledge of the attacker).

In addition to individual vulnerabilities when building the attack graph they use the attack patterns in the CAPEC format, that can act not only as input information for construction of attack graphs, but also as a result of the security analysis, as they can describe the most frequently encountered sequences of vulnerability exploitation and other actions of the attacker.

The templates also contain descriptions of attacks that do not use vulnerabilities, for example, the first stage of the attack fulfillment is to collect information about the available hosts. To do it the pattern CAPEC-292 (Host Discovery) is used, describing a group of different ways of scanning hosts and ports.

The next stage of the attack is search of vulnerable software. For this purpose the following patterns are used: CAPEC-310 (Scanning for Vulnerable Software), CAPEC-311 (Fingerprinting Remote Operating Systems), CAPEC-300 (Port Scanning), etc.

In the third stage of the attack both separate vulnerabilities from the CVE dictionary and other templates, e.g., CAPEC-233 (privilege), etc., are used.

CVSS indexes Access Vector, Access Complexity, and Authentication are the preconditions for exploiting vulnerabilities, that is, the preconditions necessary for the successful implementation of the attack. In addition to forming the graph, they are applied when evaluating the security of computer networks to determine the probability of successful attack [25], [26].

Indexes Confidentiality Impact, Integrity Impact, Availability Impact are the postconditions of vulnerabilities exploitation, that is, the postconditions of successful attack implementation. In addition to forming the graph they are applied when evaluating the computer network security to estimate damage in the result of implementation of the attack [25], [26].

Step 2. After the formation of the matrices of possible attack actions, for each host of the analyzed network the attack actions available to a specific model of the attacker based on the level of knowledge of the attacker are selected. At this stage several models of attackers can be used.

Further, on the basis of the analysis of the links of the computer network and the set of attack actions, limited by capabilities of the attacker, the graph of the availability of hosts is been formed at the same time for all attackers.

Step 3. Basing on the availability graphs, the attack trees are generated for the initial access points available to each attacker. To do this, for each attacker, the following actions are performed:

(1) formation of a set of hosts to which the attacker have access in accordance with the source data.

(2) getting the highest possible privileges on each available host, based on the use of available attack actions (by analyzing the field Gained Privileges of vulnerabilities of the group concerned).

(3) execution of attack actions, aimed at violation of the confidentiality, integrity and availability of information stored on the host. If the attacker has access only to user's rights, in accordance with the indexes Access Vector and Authentication, then the attack actions are limited to actions that are only available to local and remote users. Moreover, the impact is determined on the basis of the analysis of fields Confidentiality Impact, Integrity Impact and Availability Impact of the vulnerabilities of the group concerned.

(4) for each available host on which the attacker can get administrator's privileges the list of discovered related hosts, for which it is possible to conduct attack of information gathering, is created.

(5) compilation of the list of related hosts, for which the attacker can determine the configuration of software and hardware.

(6) the implementation of action 2 for the list formed by the action 4.

Every action of the steps of the algorithm (2-4) adds new attacking actions, that belong to the selected model of the attacker, to the attack tree.

At that for each host the directed graph of vulnerability exploitation is formed, which defines possible sequences of exploitation by the attacker. So, as a first step, the attacker can implement attacks that do not require local access and

accounts and that are targeted to:

(1) violation of the confidentiality, integrity and availability of information;

(2) obtaining access rights of the user account;

(3) obtaining access rights to the administrator account system.

Further, if the attacker gained access to any account, it can carry out attacks aimed at disrupting the confidentiality, integrity and availability of information that require local access. If the attacker has gained access to a user account, it can raise its access level to administrator with attacks for privilege elevation. Further, if the attacker has administrator's privileges, he/she can execute any attack aimed at disrupting the confidentiality, integrity and availability of information (e.g., network attacks on behalf of the account with administrator's rights).

As the result of execution of this algorithm, for each attacker the graph of connected hosts is formed, including the set of intersecting trees, starting from the initial hosts of the attacker and including the subgraphs of the exploited vulnerabilities, and attack actions aimed at collecting of information. Each host in the tree is characterized by the level of violation of the properties of confidentiality, integrity, and availability, as well as by access rights obtained by the attacker as a result of exploitation of vulnerabilities.

B. Generation of Attack Trees based on CVSS 3.0

The main disadvantages of the approach to constructing attack trees presented in the previous section are inaccuracies at using the descriptions of the vulnerabilities. For example, damage of confidentiality, integrity and availability is determined by indexes of vulnerability of the group Impact, but these indexes do not define what area of influence of vulnerability is: information in the application, information in the operating system, or all of the information on the hard disk.

Also, the constructed tree includes vulnerabilities that require active actions from the attacked host (for example, clicking on a malicious link), which is not always possible (for example, the use of such attacks is impossible against server hosts). These problems cause the necessity of transition to the standard description of vulnerabilities in CVSS of version 3.0.

The overall structure of the algorithm for constructing attack trees at transition from CVSS of version 2.0 to version 3.0 is almost the same.

But the use of CVSS of version 3.0 allows to better specify the constructed attack trees, which leads to increased validity of the constructed model and, consequently, increases the accuracy of security assessment.

Changes will be in step 1 of the algorithm presented in the previous section, as both the groups formed in this step and the values of their respective CVSS indexes will change. Consequently the results obtained in step 3 of the algorithm will change.

As the defined attack class takes into account a number of indexes (impact on confidentiality, integrity and availability, as well as the resulting privileges), the generated classes will change as well. This occurs because, first, the values of the damage changed (which will also affect the numerical assessment of the level of harm and level of risk in security assessment), and, secondly, there appeared the Scope index, on the basis of which they clarify the scope of impact of vulnerabilities (application, operating system, sandbox), and possible access to resources (files, CPU, memory, etc.) is determined.

These changes are reasons why for some attacks, where the Scope is unchanged and does not affect the system resources, the obtaining of rights will not result in obtaining the rights on the host. In addition, the scope of damage will be clarified.

Hence, the result of step 3 (items 2 and 3) of the algorithm and the list of available hosts that is generated in step 3 (item 4) of the algorithm will change.

The use of index Scope does not change the overall structure of the algorithm, but clarifies the results of the analysis of consequences of attack actions on the computer network.

At formation of the required type of access and knowledge of the attacker, in step 1 of the algorithm we use the indexes of CVSS of version 2.0 Access Vector, Authentication, and Access Complexity.

Index User Interaction, which appeared in the CVSS of version 3.0, determines whether the vulnerability may be exploited without participation of the attacker. On the basis of this index and the type of hosts, the group of vulnerabilities are outlined in the vulnerabilities matrix in step 1 of the algorithm given in section IV (A). So, the vulnerability that does not require participation of the defender, can be exploited without restrictions.

The possibility of exploitation of vulnerabilities that require the participation of the attacker, is determined on the basis of additional parameters of the host set by the operator. By default, this type of vulnerability cannot be exploited for server hosts. For custom hosts, these vulnerabilities by default are considered to be available.

For the index AccessVector the possible values have changed that, on the one hand, will affect the links in the graph, so a separate category of physical access is outlined, and the number of vulnerabilities will be removed from the graph, and, on the other hand, the value of probability of successful execution of the attacks used in the security assessment will change.

The index Authentication has been replaced by the index Privileges Required that will allow more accurately generate the list of available attack actions during the transitions between hosts and within a host, and will also affect the probability of the successful attack execution.

V. CASE STUDY AND DISCUSSION

Let us consider in an example the impact of the transition from CVSS of version 2.0 to CVSS of version 3.0 on attack tree generation and security assessment.

Fig. 1 shows an example network that contains:

- *Web-server* (with Windows Server 2008 R2 (64 bits), JBoss AS 5.0.1, ApacheStruts2 framework);
- *Database server* (with Windows Server 2008 R2 (64 bits), MS SQL Server 2008 R2, CA Spectrum 9.2, EMC Unisphere for VMAX 8.1);
- *E-mail server* (SUSE Enterprise Linux 11 SP1 (32 bits), Postfix mail server, Dovecot email server, MySQL);
- *FTP-server* (Windows Server 2008 R2 (64 bits), Ipswitch WS_FTP Server 6.1.0.0);
- Firewall-1 (Novell SUSE Linux Enterprise Server 11.0 Service Pack 3 Long Term Service Pack Support, Netfilter);
- W*orkstations* (Microsoft Windows 7 64-bit, Apple ITunes 9.0.3, Microsoft Office 2007 SP1, Microsoft Internet Explorer 7).

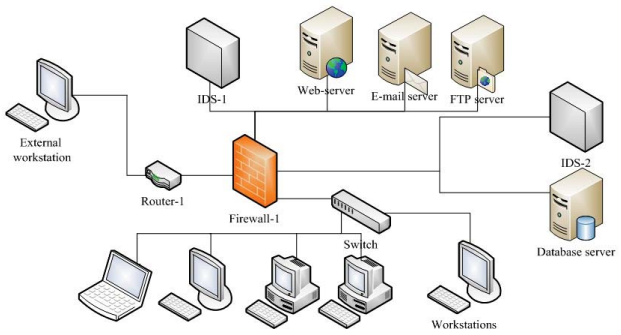

Fig. 1. Test network [25]

On the example of attacks that define a small fragment of the attack graph, we consider the impact of CVSS of version 3.0.

The remote user (the attacker) has remote access to Firewall-1, running OS Novell SUSE Linux Enterprise Server 11.0 Service Pack 3 Long Term Service Pack Support. This OS have vulnerability CVE-2016-4448. This vulnerability is assigned CVSS of version 2.0 score (10.0) and CVSS of version 3.0 score (9.8).

The appropriate CVSS of version 2.0 indexes and their values are: Access Vector “Network”, Access Complexity “Low”, Authentication “None”, Confidentiality/Integrity and Availability Impact “Complete”.

The appropriate CVSS of version 3.0 indexes and their values are: Attack Vector “Network”, Attack Complexity “Low”, Privileges Required “None”, User Interaction “None”, Scope “Unchanged”, Confidentiality/Integrity and Availability Impact “High”.

From the attack tree generation point of view, pre conditions stay the same: in both cases the necessary access level is “network”, and privileges are not required (the only difference is that in CVSS of version 3.0 it is more clear that attacker does not need additional privileges on the host), and attack complexity is low. The only difference is that in CVSS of version 3.0 it is more clear that the attacker does not need user interaction to implement an attack. And the post conditions are the same: in the both cases the attacker gets admin privileges and can proceed the attack on the next hosts and impact on the security properties is High. As soon as Scope stays unchanged, its value does not influence on the post conditions.

Next, we consider the impact of changes to CVSS of version 3.0 on the security assessment. We will present the simplified process of the attack probability calculation (without taking into account the previous attacker steps).

Attack probability for the version 2.0 was calculated using Exploitability subscore.

For the selected vulnerability it will be calculated as:

2*AccessVector*AccessComplexity*Authentication=1.0.

In case of the CVSS of version 3.0 maximum Exploitability subscore is 3.9, and minimum Exploitability subscore is 0.2.

To get value between 0 and 1.0 we subtract 0.2, divide obtained value by 10 and multiply it by 2.7.

So the probability of the successful attack action that uses the selected vulnerability in case of CVSS of version 3.0 is calculated as follows:

(8.22*AV*AC*PR*UI-0.2)*2.7/10=1.0.

The result is the same for the CVSS of version 2.0 and CVSS of version 3.0. Attack impact for the version 2.0 was calculated using Impact indexes and in consideration that only vulnerable component is impacted.

As soon as for the selected vulnerability value of the CVSS of version 3.0 Scope index is “Unchanged” impacted component is the same for both versions. Impact is also high in both cases.

But for the CVSS of version 2.0 the impact value for all security properties is 0.66, and for the CVSS of version 3.0 – 0.56. Impact subscore for the CVSS of version 2.0 is 10.0, and for the CVSS of version 3.0 – 5.9. It is not maximum value of Impact.

After Firewall compromise the attacker can discover other network hosts, for example, Database server. On this server the EMC Unisphere for VMAX 8.1 is installed. This software has vulnerability CVE-2016-6645. This vulnerability is assigned score 9.0 in CVSS of version 2.0 and the score 8.8 in CVSS of version 3.0.

The appropriate CVSS of version 2.0 indexes and their values are: Access Vector - “Network”, Access Complexity - “Low”, Authentication - “Single”, Confidentiality/Integrity and Availability Impact - “Complete”.

The appropriate indexes for CVSS of version 3.0 and their values are: Attack Vector - “Network”, Attack Complexity - “Low”, Privileges Required - “Low”, User Interaction - “None”, Scope - “Unchanged”, Confidentiality/Integrity and Availability Impact - “High”.

From the attack tree generation point of view, pre conditions stay the same: in both cases the necessary access level is “network”, and privileges are required. The only difference is that in CVSS of version 3.0 it is more clear that the attacker do need additional privileges on the host. These privileges give the basic user capabilities, which affect only the files and settings of the user.

This means that the attacker cannot directly implement the attack action, as it first needs to obtain user privileges on the host. Unlike of CVSS version 2.0 in CVSS of version 3.0 the relationship between received and required privileges became clearer.

In this example the attack complexity is low. The only difference is that in CVSS of version 3.0 it is more clear that attacker does not need user interaction to implement an attack.

Here the post conditions are the same: in the both cases attacker gets admin privileges and can proceed the attack on the next hosts, and impact on the security properties is High. As soon as Scope index value in CVSS of version 3.0 stays unchanged, its value does not influence on the post conditions.

Next, we consider the impact of changes to CVSS of version 3.0 on the security assessment.

Attack probability for the version 2.0 for the selected vulnerability will be calculated as:

2*AccessVector*AccessComplexity* Authentication=0.8.

In case of the CVSS of version 3.0 the probability of the successful attack action that uses selected vulnerability is calculated as follows:

(8.22*AV*AC*PR*UI-0.2)*2.7/10=0.7.

In this case the result for the CVSS of version 2.0 is higher than the result for the CVSS of version 3.0. As soon as for the selected vulnerability value of the index Scope is “Unchanged”, the impacted component is the same for both versions.

Impact is also high in both cases. But for the CVSS of version 2.0 the impact value for all security properties is 0.66, and for the CVSS of version 3.0 – 0.56. Impact subscore for the CVSS of version 2.0 is 10.0, and for the CVSS of version 3.0 – 5.9. It is not maximum value of Impact.

Thus, although the CVSS of version 3.0 do not fundamentally affect the algorithm of the tree building, this standard allows to remove some uncertainties and limitations.

Nevertheless it, at the same time, creates some additional complications for the process of assessment of security. An example of an additional complexity is that the Exploitability index in version 2.0 took values between 0 to 10.0, that it was easy to normalize the value of the probability of vulnerability exploitation (0 to 1.0). In version 3.0 this index takes values of 0.2 – 3.9, which is harder to be converted to possible probability values.

In addition, although the index Scope of version 3.0 allows you to separate the vulnerable component from the impacted component, it takes only two values, allowing to determine whether data components match or not. If there is a mismatch of components it remains unclear which exactly components of the system were impacted.

Our approach was implemented as a Java application. Currently full transition to CVSS of version 3.0 in the application is impossible because, first, not all vulnerabilities have the CVSS rating of version 3.0, but only new vulnerabilities do, and, second, we were not able to find .xml file with CVSS of version 3.0 data on the NVD website [21].

VII. Conclusion

In this paper we analyzed the changes introduced in the new version of the CVSS vulnerability assessment standard for, as well as the impact of these changes on the proposed algorithm for attack tree generation and security assessment. We described the refined algorithm of the attack graph generation and security assessment on the base of the CVSS of version 3.0 for the first time.

Changes in the attack tree generation and security assessment is shown by the example.

On the basis of performed analysis we concluded that the use of CVSS of version 3 will eliminate many of the ambiguities that existed previously, although not all of them.

At the moment, primarily because of the lack of the description of the complete list of the vulnerabilities using the CVSS of version 3, it is impossible to automate the application of this standard, but in the future we plan to use it in our application alongside with CVSS of version 2.0.

In addition, in further work it is planned to continue improvement of the process of the attack tree generation and security assessment from the point of view of usage of attack patterns and further automated selection of security measures.

Acknowledgment

The work is performed by the grant of RSF #15-11-30029 in St. Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences (SPIIRAS).

References

[1] M. Artz, *NetSPA, a network security planning architecture*. Master’s thesis. Massachusetts Institute of Technology, 2002, 96 p.

[2] R.P. Lippmann et al., “Validating and restoring defense in depth using attack graphs”, *in Proceedings of MILCOM 2006*, Washington, DC, pp. 1-10.

[3] K. Ingols, R. Lippmann, K. Piwowarski, “Practical attack graph generation for network defense”, *in Proceedings of 22nd Annual Conference on the Computer Security Applications*, Miami Beach, FL: IEEE, 2006, pp. 121-130.

[4] A. Singhal, X. Ou, *Security risk analysis of enterprise networks using probabilistic attack graphs*. NIST Interagency Report 7788. Gaithersburg: National Institute of Standards and Technology, 2011, 24 p.

[5] D. Man, W. Yang, Y. Yang, W. Wang, L. Zhang, "A quantitative evaluation model for network security", *in Proceedings of the 2007 International Conference on Computational Intelligence and Security*, Dec. 2007, pp. 773-777.

[6] Y.-S. Wu, B. Foo, Y.-C. Mao, S. Bagchi, E. H. Spafford, "Automated adaptive intrusion containment in systems of interacting services", *Computer Networks: The International Journal of Computer and Telecommunications Networking*, vol. 51, 2007, pp. 1334-1360.

[7] N. Stakhanova, S. Basu, J. Wong, "A cost-sensitive model for preemptive intrusion response systems", *in Proceedings of the 21st International Conference on Advanced Networking and Applications*, 2007, pp.1-8.

[8] Y. Liu, H. Man, "Network vulnerability assessment using Bayesian networks", *in Proceedings of the SPIE*, vol. 5812, 2005, pp. 61-71.

[9] M. Frigault, L. Wang, A. Singhal, S. Jajodia, "Measuring network security using dynamic Bayesian network", *in Proceedings of the ACM Workshop on Quality of Protection*, October 2008, pp. 23-30.

[10] R. Dantu, P. Kolan, J. Cangussu R. Dantu, P. Kolan, J. Cangussu, "Network risk management using attacker profiling", *Security and Communication Networks*, vol. 2, no. 1, 2009, pp. 83-96.

[11] L. Wang, T. Islam, T. Long, A. Singhal, S. Jajodia, "An Attack Graph-Based Probabilistic Security Metric", *in Proceeedings of the 22nd annual IFIP WG 11.3 working conference on Data and Applications Security*, Heidelberg: Springer-Verlag Berlin, 2008, pp. 283-296.

[12] N. Poolsappasit, R. Dewri, I. Ray, "Dynamic security risk management using Bayesian attack graphs", *IEEE Transactions on Dependable and Security Computing*, vol. 9, no. 1, 2012, pp. 61-74.

[13] I. Kotenko, A. Chechulin, "Computer Attack Modeling and Security Evaluation based on Attack Graphs", *in Proceedings of the IEEE 7th International Conference on "Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications" (IDAACS'2013)*, Berlin, Germany, September 2013, pp. 614-619.

[14] I. Kotenko, A. Chechulin, "A Cyber Attack Modeling and Impact Assessment Framework", *in Proceedings of the 5th International Conference on Cyber Conflict 2013 (CyCon 2013)*, Tallinn, Estonia: IEEE and NATO COE Publications, June 2013, pp. 119-142.

[15] I. Kotenko, E. Doynikova and A. Chechulin, "Security metrics based on attack graphs for the Olympic Games scenario", *in Proceedings of the 22th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP 2014)*, Torino, Italy, February 2014, pp. 561-568.

[16] A. Chechulin, I. Kotenko, "Attack Tree-based Approach for Real-Time Security Event Processing", *Automatic Control and Computer Sciences*, Allerton Press, Inc., vol. 49, no. 8, 2015, pp. 701-704.

[17] FIRST website. Common Vulnerability Scoring System (CVSS-SIG). Web: https://www.first.org/cvss.

[18] Common Platform Enumeration (CPE). NVD website. Web: https://nvd.nist.gov/cpe.cfm.

[19] Common Configuration Enumeration (CCE). NVD website. Web: https://nvd.nist.gov/cce/index.cfm.

[20] P. Mell, K. Scarforne, S. Romanosky, *A Complete Guide to the Common Vulnerability Scoring System (CVSS) Version 2.0*. 2007. Web: https://www.first.org/cvss/v2/guide.

[21] NVD website. Web: https://nvd.nist.gov.

[22] FIRST Org. Inc, *Common Vulnerability Scoring System v3.0: Specification Document*. 2015. Web: https://www.first.org/cvss/specification-document.

[23] S. Barnum, *Common Attack Pattern Enumeration and Classification (CAPEC). Schema Description*, 2008, 26 p.

[24] Common Vulnerabilities and Exposures (CVE). Web: http://cve.mitre.org.

[25] I. Kotenko, E. Doynikova, "Dynamical calculation of security metrics for countermeasure selection in computer networks", *in Proceedings of the 24th Euromicro International Conference on Parallel, Distributed and network-based Processing (PDP 2016)*, Heraklion, Crete, Greece, Feb. 2016. Los Alamitos, California: IEEE Computer Society, 2016, pp. 558-565.

[26] E. Doynikova, I. Kotenko, "Countermeasure selection based on the attack and service dependency graphs for security incident management", *in Proc. 10th International Conference on Risks and Security of Internet and Systems (CRiSIS 2015),* July 2015, Mytilene, Lesvos Island, Greece. Eds.: C. Lambrinoudakis and A. Gabillon, Lecture Notes in Computer Science (LNCS), vol. 9572, Springer, 2016. pp. 107-124.