Received September 16, 2017, accepted October 14, 2017, date of publication November 2, 2017, date of current version December 5, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2769063

Predicting Exploitations of Information Systems Vulnerabilities Through Attackers' Characteristics

ANDREJ DOBROVOLJC[1], DENIS TRČEK[1], AND BORUT LIKAR[2]
[1]Faculty of Computer and Information Science, University of Ljubljana, 1000 Ljubljana, Slovenia
[2]Faculty of Management, University of Primorska, 6000 Koper, Slovenia

Corresponding author: Andrej Dobrovoljc (andrej.dobrovoljc@3ad.si)

The work of A. Dobrovoljc was supported in part by the European Union through the European Social Fund, Operational Program for Human Resources Development for the period 2007–2013 and in part by Slovenian Research Agency under Project N1-0057.

ABSTRACT The main goal of proactive security is to prevent attacks before they happen. In modern information systems it largely depends on the vulnerability management process, where prioritization is one of the key steps. A widely used prioritization policy based only upon a common vulnerability scoring system (CVSS) score is frequently criticised for bad effectiveness. The main reason is that the CVSS score alone is not a good predictor of vulnerability exploitation in the wild. Therefore, the aim of the research in this field is to determine in what way we can improve our prediction abilities. Clearly, software vulnerabilities are commodities used by attackers. Hence, it makes sense considering their characteristics in vulnerability prioritization. In contrast, one should be able to measure and compare the effectiveness of various policies. Therefore, an important goal of this paper was to develop an evaluation model, which would allow such comparisons. For this purpose, we developed an agent-based simulation model which measures the exposure of information system to exploitable vulnerabilities. Besides, some policies which take into account human threats were defined and then compared with the most popular existing methods. Experimental results imply that the proposed policy, which is based on CVSS vectors and attacker characteristics, achieves the highest effectiveness among existing methods.

INDEX TERMS CVSS, prioritization policy, security management, threat agent, vulnerability.

I. INTRODUCTION

Vulnerabilities in software represent a serious risk for contemporary information systems (IS). According to the National Vulnerability Database (NVD),[1] which maintains records of all acknowledged vulnerabilities of software products on the market, more than 90,000 vulnerabilities have been discovered since 1997. Secunia Research [1], which applies different approach to recording, reported 17,147 new vulnerabilities in 2016. This number shows a 33% increase in the five year trend. These high numbers prove that discovering vulnerabilities has become an extremely widespread activity and therefore, new threats are constantly being faced.

So as to maintain IS secure and to keep the costs of security within limited budgets, companies are forced to manage risks by applying various approaches. Since each vulnerability presents different level of threat, it is necessary to apply an appropriate prioritization policy [2], [3].

Although very hard to achieve, a quantitative risk assessment and a unified evaluation process are preferred to make comparisons among security issues. Therefore, the National Infrastructure Advisory Council (NIAC) introduced an open standard Common Vulnerability Scoring System (CVSS) in 2005, which allows quantitative assessment of vulnerabilities' severity [4]. CVSS is widely accepted today and has become an integral part of automated vulnerability management tools based on SCAP protocol[2] [5], [6].

Although CVSS represents an important contribution to the security management area, it is still largely unexplored as whether CVSS scores are accurate and the most suitable for prioritization purposes [7]. In addition, the danger represented by vulnerability CVSS score does not match the risk of exploitation in the wild [6]. However, organizations need a clear answer as regards the most effective vulnerability prioritization policy for their IS [8].

In order to improve prediction abilities, it is critical to identify features that can be observed as indicators of potential exploitations [9]. For instance, it is known that incorporating attacker characteristics in risk estimation improves its

[1]https://nvd.nist.gov/

[2]https://scap.nist.gov/

accuracy [10]. Therefore, we assume that taking into consideration attacker's characteristics improves vulnerability prioritization. Our intention is to discover alternative methods of prioritization, which would be more effective in suppressing attacks. Our study does not address the checking of CVSS scores accuracy.

To our knowledge, currently there is no general solution, which would allow assessment of effectiveness and comparisons among various vulnerability prioritization policies. Therefore, an important step of our study was to develop such a method.

The main contributions of this paper are:

- A model for evaluating the effectiveness of vulnerability prioritization policies is proposed. For this purpose a metric measuring the exposure of information system to exploitable vulnerabilities was defined.
- A vulnerability prioritization policy, which is based on CVSS vectors and attacker characteristics, is presented, and which assures faster risk reduction than policy based only on a CVSS score.
- We show that the number of threat agents, who can exploit a vulnerability, is a significantly better predictor of exploitable vulnerabilities than CVSS score.

The rest of the paper is organized as follows. Section II presents the related work. Section III explains the model for comparing the effectiveness of various vulnerability prioritization policies. Section IV describes the evaluation and presents results of the experiment. In section V, our findings are discussed. Paper concludes with Section VI where some future plans are presented.

II. RELATED WORK

CVSS is a universal and vendor-independent scoring system for quantitative measurement of severity of software vulnerabilities. One of its main purposes is to prioritize vulnerabilities for neutralization according to the risk they present [4]. It also represents the basis for several risk assessment models [11]–[13].

CVSS is based on three groups of attributes and equations for calculating scores (Base, Temporal and Environmental). Values of Base attributes and scores are publicly available in NVD and represent vulnerability intrinsic characteristics. Temporal attributes provide information on vulnerability changes over time. They can be specified by security analysts and vendors, yet are rare in practice. Environmental attributes represent the contextual information on an environment and can be provided only by an IS owner [4].

CVSS has been criticised for several things. The distribution of Base score is highly bimodal and many combinations of attributes produce the same final score [4]. There are also doubts about the accuracy of scores [7], [14]. Vulnerability Rating and Scoring System (VRSS), which is an alternative system based on CVSS attributes, achieves higher diversity of scores [14], [15]. However, there is no evidence that VRSS scores are more representative than CVSS scores [7]. Besides, there are several types of bias that impact the data in vulnerability databases, which suppress accurate statistical analysis [16].

It is frequently pointed out in literature that CVSS Base score, when applied alone, is not suitable for vulnerability prioritization purposes [5], [17]. The most extreme claim says that using CVSS score is almost as effective as a random approach [6]. The reason for such conclusions is the fact that many vulnerabilities with a high severity score are never exploited in the wild [17].

A good risk factor, which can be used for making predictions about future exploitations, is the information about presence of exploit in the black markets [6]. Unfortunately, CVSS Temporal data, which are supposed to provide such information and would allow good predictions [3], are not available in NVD. They exist in limited sizes and in various forms on vendor sites and other services, making them inconvenient for end users. Indirectly, the need for Temporal data was also expressed by security experts in an experiment conducted by Holm and Khan [7].

In such situation, alternative methods are needed to improve proactive security. It may be achieved on the development side by way of predicting vulnerable software components [18], [19] or predicting the number of vulnerability discoveries in the future [20]. However, more critical is a deployment part, where the risk estimation of already known vulnerabilities is required. The authors of CVSS standard try to improve its comprehensiveness by providing better descriptions of vulnerabilities. Each new release of CVSS standard has introduced some major changes and additions within the set of attributes [21], [22].

Nonetheless, numerous other factors strongly impact the risk and are unaccounted yet with the existing approaches. One of them is assessing risk with respect to adversary intent and capabilities, since there is a human agent with a motivation behind each attack [23]. Therefore, we have assumed that the system security may be improved by considering the attacker characteristics in vulnerability prioritization.

III. A MODEL FOR EVALUATION OF VULNERABILITY PRIORITIZATION POLICIES

This section is divided into five subsections. Subsection III-A presents the idea for considering attacker characteristic in vulnerability prioritization, Subsections III-B, III-C and III-D define key variables for evaluation of vulnerability exploitation by threat agents, and Subsection III-E presents the evaluation model for measuring the effectiveness of prioritization policies.

A. CONSIDERING ATTACKER CHARACTERISTICS IN VULNERABILITY PRIORITIZATION

Vulnerabilities have to be prioritized according to the risk they represent for IS. The question is whether a higher CVSS Base score always means also a higher risk. Equation (1) shows that the security risk depends on a number of factors [24], whereat in this case threats deserve special attention. Namely, vulnerabilities are goods used by attackers and

TABLE 1. Description of vulnerability CVSS Base attributes (CVSSv2 standard).

Attribute name	Description and values
Access Vector	Reflects how the vulnerability can be exploited by an attacker. `NETWORK, ADJACENT NETWORK, LOCAL`
Access Complexity	Defines the complexity of an attack after gaining access. `LOW, MEDIUM, HIGH`
Authentication	Defines the number of times to authenticate to exploit the vulnerability. `NONE, SINGLE INSTANCE, MULTIPLE INSTANCES`
Confidentiality Impact	Defines the impact on confidentiality (disclosing information, limiting and preventing access). `NONE, PARTIAL, COMPLETE`
Integrity Impact	Defines the impact on veracity of information. `NONE, PARTIAL, COMPLETE`
Availability Impact	Defines the impact on accessibility of IS resources. `NONE, PARTIAL, COMPLETE`

therefore, appertain to human types of threats.

$$Risk = f(AssetValue, Threat\ Probability, Vulnerability, Impact) \quad (1)$$

So as to carry out a successful attack an attacker ought to have three things: methods (skills, knowledge, tools, etc.), opportunity and motivation [25]. An important influencing factor is also the expected impact as it is seen from (2) [26]. Based on these facts it has been concluded that the exploitation of vulnerability depends on attackers' characteristics. Simply said, when two vulnerabilities have equal CVSS Base score, the one which is exploitable by more attackers represents a higher risk.

$$Threat = f(Capability, Opportunity, Motivation, Expected\ Impact) \quad (2)$$

However, some of the vulnerabilities with high CVSS scores are not even exploitable in the wild. Instead of merely observing CVSS Base score, it is better to find alternative approaches for making better predictions of future exploitations.

We reasonably assume that the effectiveness of vulnerability prioritization may be improved by taking into consideration the attacker characteristics. The study further assumes that vulnerability with a greater number of potential attackers is more likely to be exploited. The question is how to define the relationship between the attacker's characteristics and the properties of vulnerability in regard to exploitation. The second important question is how to measure the effectiveness of a policy and compare it with others.

The main challenge is to define the borderline requirements of a potential attacker seeking exploitation of an individual vulnerability. For this purpose, detailed information on vulnerability and attacker is required.

According to CVSS standard, each vulnerability is described with a CVSS vulnerability vector. This is a group of qualitative attributes, which provide basic information on opportunities for exploitation, required attacker capabilities, and potential impact. Currently available data in NVD appertain to CVSSv2 release of the standard, which defines attributes according to Table 1. CVSSv3 vulnerability data is only available a short time and slightly upgrades the attribute model.

Similarly, there are some threat libraries, which define typical groups of attackers. One such initiative, suitable for our purposes is Threat Agent Library (TAL) [27]. Threat agents are described by eight attributes given in Table 2, where all except two (*Outcome* and *Objectives*) are defined by ordered sets of values. The values are ordered from the least to the most serious.

TAL defines 21 threat agent archetypes, which according to the authors' opinion represent all typical groups of attackers in common environment. At this point, the exploitation condition has to be defined in order to find out, whether individual threat agent from this library is a candidate to exploit a vulnerability with a given CVSS vector. For this purpose we define the following Boolean function:

$$isExploited\ (ThreatAgent,\ CVSS\ Vector) = capability \wedge opportunity \wedge motivation \quad (3)$$

Function *isExploited* follows the definition of a human threat, which is given by (2). It depends on characteristics of a threat agent as well as on properties of vulnerability. However, for prioritization purposes we need a total number of threat agents from TAL, who can exploit a vulnerability with a given CVSS vector. Therefore, they are counted in variable *TAC* (Threat Agent Count). It is defined by (4), where `TRUE` values are evaluated as 1 and `FALSE` as 0.

$$TAC = \sum_{agent \in TAL} isExploited\ (agent, vect) \quad (4)$$

According to our expectations, it is first necessary to remove vulnerabilities with a greater number of threat agents so as to reduce the greatest risks. In our case, it means that

TABLE 2. Description of Threat Agent attributes defined by TAL library.

Attribute name	Description and values
Intent	Defines whether the agent intends to cause harm (ordinal values). `NON HOSTILE < HOSTILE`
Skills	Defines the expertise an agent possesses (ordinal values). `NONE < MINIMAL < OPERATIONAL < ADEPT`
Resources	Defines the organizational level at which an agent typically works (ordinal values). `INDIVIDUAL < CLUB < CONTEST < TEAM < ORGANIZATION < GOVERNMENT`
Access	Defines the type of the agent's access to the assets (ordinal values). `EXTERNAL < INTERNAL`
Limits	Defines the extent to which the agent is prepared to break the law (ordinal values). `CODE OF CONDUCT < LEGAL < EXTRA LEGAL MINOR < EXTRA LEGAL MAJOR`
Visibility	Defines the extent to which the agent intends to conceal or reveal his or her identity (ordinal values). `CLANDESTINE < COVERT < OVERT < MULTIPLE`
Outcome	Defines what the agent hopes to accomplish (multiple values allowed). `ACQUISITION THEFT, BUSINESS ADVANTAGE, DAMAGE, EMBARRASSEMENT, ENTERTAINMENT, TECH ADVANTAGE`
Objectives	Defines the action that the agent intends to take (multiple values allowed). `COPY, DENY, DESTROY, DAMAGE, TAKE, ALL`

vulnerability with the highest TAC value is removed first. If two vulnerabilities have equal TAC values, the first found is removed first. We call this policy HTAC (Highest Threat Agent Count), which corresponds to the method of selecting vulnerabilities to be removed.

In the following subsections, the definitions of variables *capability*, *opportunity* and *motivation* from function *isExploited* are specified.

B. CAPABILITY FOR VULNERABILITY EXPLOITATION

At first a threat agent needs sufficient level of capability to be successful at vulnerability exploitation. Capability depends on its skills and resources. Therefore, in (5) we defined the variable *agentCapability*, which is the product of ordinal values of agent's *Skills* and *Resources* attributes. Ordinal values are given in Table 3 next to the attributes. *Skills* has 4 and *Resources* has 6 different values, which produce 24 domain values of variable *agentCapability*.

$$\begin{aligned} agentCapability = {} & Ord(Agent.Skills) \\ & * Ord(Agent.Resources) \end{aligned} \tag{5}$$

Vulnerabilities differ in their complexity, which is determined through the *Access Complexity* CVSS attribute (Table 1). It defines three levels of complexity: LOW, MEDIUM and HIGH. Threat agents with higher capability are capable of exploiting more complex vulnerabilities. Therefore, we divided *agentCapability* domain values into three equally large intervals, with 8 values in each interval. Higher value represents higher agent capability. Finally, the Boolean *capability* variable from (3) is calculated according to the conditions in the Table 4.

Allow us to take an example with the following CVSS vulnerability vector, which is one of the most frequent ones in the NVD database: [AV:N/AC:M/Au:N/C:P/I:P/A:P]

From AC:M it may be recognized that the value of *Access Complexity* attribute is MEDIUM. Therefore, *agentCapability* must be greater than 8 for successful exploitation (see Table 4). Let us calculate now the *capability* variable for Data Miner threat agent. It has *Agent.Skills* $=$ ADEPT and *Agent.Resources* $=$ TEAM, which results in the following calculation:

$$\begin{aligned} capability &= Ord(\mathtt{ADEPT}) * Ord(\mathtt{TEAM}) > 8 \\ &= 4 * 4 > 8 = True \end{aligned} \tag{6}$$

It was found that Data miner possesses enough capability to exploit CVSS vector [AV:N/AC:M/Au:N/C:P/I:P/A:P].

C. AN OPPORTUNITY OFFERED BY VULNERABILITY

In general vulnerabilities are opportunities for threat agents to exploit the IS. However, various constraints exist on both sides and determine whether vulnerability may be exploited by a threat agent. Therefore, the Boolean variable *opportunity* is calculated in our model according to (7) from the following three components: *accessOpportunity*, *authOpportunity* and *objectiveOpportunity*. All three conditions have to be met for a threat agent to get sufficient opportunity of vulnerability exploitation.

$$\begin{aligned} opportunity = {} & accessOpportunity \wedge authOpportunity \\ & \wedge objectiveOpportunity \end{aligned} \tag{7}$$

The most obvious is the constraint of access to vulnerability. CVSS distinguishes three difficulty levels (Table 1). Vulnerabilities, which are exploitable over the internet, allow for the easiest access (*Access Vector* $=$ NETWORK). Everyone has an opportunity to do it on its own. Hence, the value of *accessOpportunity* variable is TRUE for all threat agents in this case. The hardest thing is to access vulnerabilities,

TABLE 3. Threat Agent Library (TAL) archetypes and their attributes. Next to the attributes are corresponding numeric ordinal values, which were used in evaluation (adapted from [27]).

		Employee Recless	Employee Untrained	Info Partner	Anarchist	Civil Activist	Competitor	Corrupt Government Official	Data Miner	Employee Disgruntled	Government Cyberwarrior	Government Spy	Internal Spy	Irrational Individual	Legal Adversary	Mobster	Radical Activist	Sensationalist	Terrorist	Thief	Cyber Vandal	Vendor
Intent	Non hostile (1)	*	*	*																		
	Hostile (2)				*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
Access	Internal (1)	*	*	*						*		*	*							*		*
	External (2)				*	*	*	*	*		*			*	*	*	*	*	*		*	
Visibility	Clandestine (1)			*			*		*			*	*							*		*
	Covert (2)	*				*										*			*		*	
	Overt (3)		*		*			*							*		*	*				
	Multiple (4)									*	*			*								
Resources	Individual (1)	*	*	*						*				*						*		
	Club (2)				*													*				
	Contest (3)																				*	
	Team (4)								*													*
	Organization (5)					*	*						*		*	*	*		*			
	Government (6)							*			*	*										
Skills	None (1)				*									*						*		
	Minimal (2)		*															*				
	Operational (3)			*						*											*	*
	Adept (4)	*				*	*	*	*		*	*	*		*	*	*		*			
Limits	Code of Conduct (1)		*	*																		
	Legal (2)	*													*							*
	Extra-legal, minor (3)					*	*	*	*				*				*	*		*	*	
	Extra-legal, major (4)				*					*	*	*		*		*			*			
Objectives	Copy					*	*		*			*	*		*							*
	Deny							*			*											
	Destroy				*					*	*								*			
	Damage									*	*								*			
	Take														*	*			*	*		
	All of the Above	*	*	*										*			*	*			*	
Outcome	Acquisition/ Theft												*			*				*		
	Business Advantage						*	*	*			*			*							*
	Damage	*	*	*	*					*	*			*			*	*	*		*	
	Embarrassment	*	*	*		*				*	*			*	*		*	*				
	Tech Advantage						*	*	*			*	*									*

which are exploitable only on a local computer (*Access Vector* = LOCAL). In this case, the opportunity is narrowed down only to threat agents with an internal access to vulnerable component. There are vulnerabilities between these extremes. These vulnerabilities are exploitable from adjacent networks (*Access Vector* = ADJACENT NETWORK). We define that such vulnerabilities represent sufficient opportunity for threat agents, who can get help from the others. Conditions to calculate Boolean variable *accessOpportunity* are given in Table 5.

TABLE 4. Calculation of Boolean capability variable for different values of Access Complexity attribute.

CVSS Access Complexity	capability
LOW	True
MEDIUM	$agentCapability > 8$
HIGH	$agentCapability > 16$

TABLE 5. AccessOpportunity conditions for corresponding Access Vector attribute values.

CVSS Access Vector	accessOpportunity
LOCAL	$Agent.Access =$ INTERNAL
ADJACENT NETWORK	$Agent.Resources >$ INDIVIDUAL
NETWORK	True

In case of vector [AV:N/AC:M/Au:N/C:P/I:P/A:P], the value of *accessOpportunity* is TRUE, due to vulnerability *Access Vector* is NETWORK (AV:N). Therefore, Data Miner recognises this vector as an opportunity to access the target.

In many cases an authentication to the system is required prior to actual exploitation of vulnerability. As known every authentication can leave traces. Therefore, exploitation of such vulnerability is not an opportunity for anyone. Threat agents differ in their *Visibility* attribute, which sets forth to what extent they are prepared to reveal their identity. CVSS specifies three values of vulnerability *Authentication* attribute. However, our model distinguishes only two types of threat agent behaviour. Vulnerabilities, which do not require any authentication (*Authentication* = NONE), represent opportunity for everybody. All the other vulnerabilities are opportunities only for threat agents, who are not concerned about disclosure of their identity. Conditions are defined in Table 6.

TABLE 6. AuthOpportunity conditions for corresponding Authentication attribute values.

CVSS Authentication	authOpportunity
NONE	True
SINGLE INSTANCE	$Agent.Visibility \geq$ OVERT
MULTIPLE INSTANCES	$Agent.Visibility \geq$ OVERT

CVSS vector [AV:N/AC:M/Au:N/C:P/I:P/A:P] shows that vulnerability *Authentication* attribute value is NONE (Au:N). Such a vulnerability is an opportunity for anyone (see Table 6). Therefore, the value of *authOpportunity* is TRUE. Data miner reveals the absence of authentication as an opportunity.

The third opportunity component is *objectiveOpportunity*. Vulnerability has to give the threat agent an opportunity to achieve his/her specific objectives. Threat agent objectives are enumerated in Table 2. Each action to achieve objective threatens the IS in at least one security context: confidentiality (C), integrity (I) or availability (A). In (8) threat agent *Objective* attributes from Table 2 are grouped into corresponding sets C, I and A.

$$\begin{aligned} C &= \{COPY, TAKE, ALL\} \\ I &= \{DENY, DAMAGE, ALL\} \\ A &= \{DESTROY, TAKE, ALL\} \end{aligned} \tag{8}$$

On the other hand the CVSS vulnerability vectors denote, which security contexts can be threatened by exploitation of a vulnerability (Impact attributes in Table 1). When vulnerability impact and threat agent objective appertain to the same context, vulnerability represents an opportunity for a threat agent. This is defined by (9).

$$\begin{aligned} &objectiveOpportunity \\ &= (Vuln.ConfImpact \neq \text{NONE} \wedge Agent.Objectives \in C) \\ &\quad \vee (Vuln.IntegImpact \neq \text{NONE} \wedge Agent.Objectives \in I) \\ &\quad \vee (Vuln.AvailImpact \neq \text{NONE} \wedge Agent.Objectives \in A) \end{aligned} \tag{9}$$

The CVSS vector [AV:N/AC:M/Au:N/C:P/I:P/A:P] reveals that vulnerability has an impact on confidentiality, integrity and availability. All *Impact* attributes have value PARTIAL (C:P/I:P/A:P). Data Miner's objective is COPY, which appertains to the confidentiality set C. Based on (9) we conclude that *objectiveOpportunity* variable is TRUE, since vulnerability and Data Miner have confidentiality context in common.

At this point it was discovered that all opportunity criteria in our example are met and therefore the Boolean variable *opportunity* is TRUE. Hence, it was concluded that Data Miner recognizes the CVSS vulnerability vector [AV:N/AC:M/Au:N/C:P/I:P/A:P] as an opportunity.

D. MOTIVATION FOR VULNERABILITY EXPLOITATION

Threat agent motivation for vulnerability exploitation depends on several attributes. Therefore, we define a Boolean *motivation* variable as a combination of three components: hostility, desired outcome and meeting the legal and ethical limits:

$$\begin{aligned} motivation &= motiveHostile \\ &\quad \wedge motiveOutcome \wedge motiveMeetLimits \end{aligned} \tag{10}$$

Vulnerability exploitation is a domain of hostile threat agents only. Their intention is to cause damage or take advantage in an unjust way. They are recognized by threat agent *Intent* attribute. *motiveHostile* is calculated as follows:

$$motiveHostile = (Agent.Intent = \text{HOSTILE}) \tag{11}$$

Motivated threat agent has at least one outcome in his/her mind. Therefore, motivation to achieve something hostile may be determined by the threat agent *Outcome* attribute. *motiveOutcome* is calculated as follows:

$$motiveOutcome = (Agent.Outcome \neq \{\}) \tag{12}$$

Some threat agents have legal and ethical limits by way of which the extent to which they are prepared to break the law is defined. For this purpose TAL sets forth *Limits* an attribute with four different values. They are presented in Table 2. Threat agents are interested in using only vulnerability vectors according to their needs and limitations. For example, agents who strictly comply with the law are not interested in exploitation of any vulnerability (*Agent.Limits* = `CODE OF CONDUCT`). On the other hand, threat agents without limitations do not hesitate to exploit any vulnerability (`EXTRA LEGAL MAJOR`).

Amid these extremes there are threat agents with `LEGAL` limitations. They act within the limits of applicable laws. Therefore, they are interested in vulnerabilities which allow legally permitted activities. In line with our interpretation, only vulnerability vectors with *Confidentiality impact* attribute value `PARTIAL` meet up this condition (`C:P`). Such threat agents are interested in reading specific information yet do not cause direct harm.

Threat agents with `EXTRA LEGAL MINOR` limitations may break the law in relatively minor, non-violent ways, such as minor vandalism or trespass. Therefore, they are interested only in vulnerabilities which allow for such activities. Consistent with our interpretation, they are interested in vulnerability vectors with `PARTIAL` impact in either security context (confidentiality or integrity or availability) as well as in vectors with `COMPLETE` confidentiality impact (`C:P or I:P or A:P or C:C`). Following our opinion, public disclosure of confidential information does not represent such a huge offense as destruction or damage.

TABLE 7. Allowed vulnerability vectors (*V*) according to the threat agent Limits attribute.

Agent.Limits	*V* (allowed are vectors with the following CVSS Impact attribute values)
`CODE OF CONDUCT`	None
`LEGAL`	C:P
`EXTRA LEGAL MINOR`	C:C or C:P or I:P or A:P
`EXTRA LEGAL MAJOR`	Any

Boolean variable *motiveMeetLimits* is calculated according to (13), where *cvssVect* is a vulnerability vector to exploit and V is a set of allowed vectors according to threat agent limitations. They are defined in Table 7.

$$motiveMeetLimits = (cvssVect \in V) \tag{13}$$

Calculating now the Boolean *motivation* variable for vulnerability vector `[AV:N/AC:M/Au:N/C:P/I:P/A:P]` and Data Miner threat agent the result is: variable *motiveHostile* is `TRUE`, because Data Miner is a hostile agent (*Agent.Intent* = `HOSTILE`). Variable *motiveOutcome* is `TRUE`, because Data Miner has two *Outcome* values (Business Advantage and Tech Advantage). Variable *motiveMeetLimits* is `TRUE` as well, because Data Miner limitation is `EXTRA LEGAL MINOR` and the vulnerability vector appertains to allowed vectors according to Table 7 (all three impact attribute values `C:P`, `I:P` and `A:P` are of interest for Data miner). We have concluded that Data Miner is motivated for exploitation of vulnerability vector `[AV:N/AC:M/Au:N/C:P/I:P/A:P]`, given that all motivation conditions are met.

Now *isExploited* function may be evaluated. Vulnerability vector `[AV:N/AC:M/Au:N/C:P/I:P/A:P]` is exploitable by Data Miner because it is capable to exploit it (*capability* = `TRUE`), vulnerability represents opportunity for the agent (*opportunity* = `TRUE`) and agent is motivated for vulnerability exploitations (*motivation* = `TRUE`).

E. MODEL FOR EVALUATING THE EFFECTIVENESS OF VULNERABILITY PRIORITIZATION POLICIES

In order to verify hypotheses, a suitable measurement instrument has to be developed, which would allow for quantitative comparisons of effectiveness of different vulnerability prioritization policies. According to our knowledge, there is no such method, which would allow such comparisons. The existing studies mainly dealt with the accuracy of CVSS scores and their usefulness.

An effective vulnerability prioritization policy has to fix vulnerabilities according to real danger they represent for the IS. In reality, only a part of vulnerabilities is exploitable at any moment. Fixing vulnerabilities just because of high CVSS score is not the most efficient approach, because this score is simply not a good predictor of exploitations in the wild. Significantly better results may be achieved by fixing vulnerabilities, for which proof-of-concept exploits exist, or even better in response to exploit presence in the black market [6].

The most effective policy has to remove exploitable and very likely exploitable vulnerabilities first, since they represent the highest risk for IS. However, information on the existence of exploits is not always available and in secrecy constantly appear new exploits. Therefore, in order to improve the effectiveness of vulnerability prioritization, methods for making predictions about likely future exploitations are needed. The lower the number of exploitable vulnerabilities in IS, the lower the exposure of IS to threat is. Therefore, a good method to compare the effectiveness of policies is to measure the exposure of IS to exploitable vulnerabilities.

Based on the latter fact, a metric for comparing the effectiveness of policies may be defined. The number of exploitable vulnerabilities in IS changes over time since vulnerability with the highest priority is removed with each step of policy application. Let the set of remaining exploitable vulnerabilities in IS on *i-th* step be E_i. Then the exposure of IS to exploitable vulnerabilities in time frame $[0..k]$ can be defined as presented in (14), where k is the number of times the policy has been used to remove vulnerabilities from the IS. According to this definition, the lower value of $ExposureEV_k$

represents higher policy effectiveness.

$$ExposureEV_k = \sum_{i=0}^{k} |E_i| \tag{14}$$

An experiment will be carried out to evaluate the proposed metric as to individual policy. For this purpose, existing credible data is required. Vulnerability data may be found in various data sources. We are interested in data on all discovered vulnerabilities in the observed period as well as in data on their exploitation in practice. To get a complete picture of each vulnerability, all these data must be integrated into a common database.

A complete list of all discovered vulnerabilities with their properties is available in the NVD database. Unfortunately, similar database of all exploited vulnerabilities does not exist. There are some datasets that can be used to draw conclusions on the exploitation of individual vulnerabilities. One such database, which is often used for research purposes, is EDB (Exploit-db).[3] It is an archive of all sorts of exploits. Furthermore, it is CVE compliant (CVE, Common Vulnerabilities and Exposures). This database also represents the basis for penetration testing. Among other resources, we need to mention specific vulnerability lists that are related to certain groups of exploitation. One such resource is the list of vulnerabilities used in exploitation kits and available on the black market (EKITS).[4] For the purpose of our experiment, data from NVD, EDB and EKITS databases have been used.

Firstly, the interest was focused in the effectiveness of the proposed HTAC policy and its comparison with the other existing methods. Another interesting question is how much these policies are better than the random order of removal. However, we are also interested in prediction abilities of each method in regard to exploitations. Therefore, the experiment shall statistically analyse, how fast individual methods can remove exploitable vulnerabilities.

IV. EVALUATION

Evaluation of the proposed model took place in three steps:

- Calculation of TAC values of all CVSS vectors and all vulnerabilities used in the experiment.
- Comparison of the effectiveness of policies.
- Evaluation of policy prediction abilities.

The CVSS vector is composed of 6 attributes, which produce 729 different combinations altogether. In Table 8, the left most column presents all TAC values, based on 21 threat agents from TAL. TAC never exceeded the value 18. The "*CVSS Vect.*" column represents the numbers of CVSS vectors, with the same TAC value. Table 9 presents the number of CVSS vectors exploitable by individual threat agents from TAL.

In the experiment, NVD database in the period from 2010 to 2016 was used. In total, there were 41,823 vulnerabilities.

[3]http://cve.mitre.org/data/refs/refmap/source-EXPLOIT-DB.html

[4]http://contagiodump.blogspot.si/2010/06/overview-of-exploit-packs-update.html

TABLE 8. Number of CVSS vectors and vulnerabilities in NVD, EDB and EKITS databases exploitable by the same number of threat agents (TAC value).

TAC	CVSS Vect.	NVD	EDB	EKITS
18	4	4,085	793	4
17	4	401	10	
16	4	3,257	137	41
15	8	49	2	
14	5	2,361	162	1
13	16	5,266	272	
12	10	226	16	1
11	17	4,951	216	44
10	18	2,174	75	
9	18	434	22	1
8	18	1,636	96	1
7	25	702	24	
6	29	1,608	33	
5	40	6,235	314	1
4	82	3,342	131	1
3	120	2,265	54	
2	74	1,506	47	
1	86	743	9	
0	151	582	3	
Sum	729	41,823	2,416	95

TABLE 9. Threat Agents with the number of favorable CVSS Vectors.

Agent Name	Number of CVSS Vectors
Employee Recles	0
Employee Untrained	0
Info Partner	0
Anarchist	108
Civil Activist	108
Competitor	108
Corupt Government Officer	288
Data Miner	72
Employee Disgruntled	144
Government Cyberwarrior	432
Government Spy	162
Internal Spy	162
Irrational Individual	78
Legal Adversary	162
Mobster	144
Radical Activist	414
Sensationalist	138
Terrorist	156
Thief	44
Cyber Vandal	92
Vendor	54

The number of vulnerabilities from NVD with the same TAC value are presented in Table 8 in *NVD* column. Among them, there are 2,416 vulnerabilities, for which proof-of-concept (PoC) exploits exist in EDB database (*EDB* column). Therefore, they are called PoC vulnerabilities. Moreover, 95 exploitable vulnerabilities from the same period were also evaluated. All these vulnerabilities are exploited in the wild and they are recorded in the EKITS database (*EKITS* column).

The effectiveness of the policy is much more dependent on frequent CVSS vectors than on the less frequent ones. In addition, the CVSS vectors that occur frequently in EDB deserve greater attention than less frequent ones. Therefore, both databases were evaluated in terms of CVSS vector frequency as well as their TAC values and CVSS scores. The most frequent CVSS vectors from NVD database are listed in Table 10, the most frequent CVSS vectors from EDB in Table 11 and vectors from EKITS in Table 12. The tables in *%PoC* column represent the share of NVD vulnerabilities based on the current CVSS vector for which a PoC exploit in EDB database exists.

TABLE 10. CVSS vectors with more than 1,000 vulnerabilities in NVD with associated TAC value, CVSS score and the share of PoC vulnerabilities.

Cvss Vector	Count	TAC	CVSS	% PoC
AV:N/AC:M/Au:N/C:N/I:P/A:N	4,764	5	4.3	5.79%
AV:N/AC:L/Au:N/C:P/I:P/A:P	4,052	18	7.5	19.57%
AV:N/AC:M/Au:N/C:C/I:C/A:C	3,672	11	9.3	5.45%
AV:N/AC:L/Au:N/C:C/I:C/A:C	3,229	16	10	4.24%
AV:N/AC:M/Au:N/C:P/I:P/A:P	3,213	13	6.8	8.22%
AV:N/AC:L/Au:N/C:P/I:N/A:N	2,353	14	5	6.88%
AV:N/AC:L/Au:N/C:N/I:N/A:P	1,967	10	5	3.20%
AV:A/AC:M/Au:N/C:P/I:P/A:P	1,423	13	5.4	0.00%
AV:L/AC:L/Au:N/C:C/I:C/A:C	1,103	4	7.2	6.71%
AV:N/AC:M/Au:N/C:N/I:N/A:P	1,056	5	4.3	2.84%
AV:N/AC:M/Au:N/C:P/I:N/A:N	1,050	11	4.3	0.76%
Sum	27,882			

TABLE 11. CVSS vectors with more than 30 vulnerabilities in EDB with associated TAC value, CVSS score and the share of PoC vulnerabilities.

CVSS Vector	Count	TAC	CVSS	% PoC
AV:N/AC:L/Au:N/C:P/I:P/A:P	793	18	7.5	19.57%
AV:N/AC:M/Au:N/C:N/I:P/A:N	276	5	4.3	5.79%
AV:N/AC:M/Au:N/C:P/I:P/A:P	264	13	6.8	8.22%
AV:N/AC:M/Au:N/C:C/I:C/A:C	200	11	9.3	5.45%
AV:N/AC:L/Au:N/C:P/I:N/A:N	162	14	5	6.88%
AV:N/AC:L/Au:N/C:C/I:C/A:C	137	16	10	4.24%
AV:N/AC:L/Au:S/C:P/I:P/A:P	79	8	6.5	10.45%
AV:L/AC:L/Au:N/C:C/I:C/A:C	74	4	7.2	6.71%
AV:N/AC:L/Au:N/C:N/I:N/A:P	63	10	5	3.20%
AV:L/AC:M/Au:N/C:C/I:C/A:C	34	2	6.9	5.63%
Sum	2,082			

In the second step of evaluation, the effectiveness of several policies was compared. They are listed in Table 13. CVSS gives priority to vulnerabilities with the highest CVSS Score. Its calculation is defined in [21]. Similarly, the VRSS method gives priority to vulnerabilities with the highest VRSS Score. The VRSS Score calculation is defined in [14]. The FIFO

TABLE 12. CVSS vectors of vulnerabilities used by EKITS with associated TAC value, CVSS score and the share of PoC vulnerabilities.

CVSS Vector	Count	TAC	CVSS	% PoC
AV:N/AC:M/Au:N/C:C/I:C/A:C	43	11	9.3	5.45%
AV:N/AC:L/Au:N/C:C/I:C/A:C	41	16	10	4.24%
AV:N/AC:L/Au:N/C:P/I:P/A:P	4	18	7.5	19.57%
AV:L/AC:L/Au:N/C:C/I:C/A:C	1	4	7.2	6.71%
AV:N/AC:H/Au:N/C:C/I:C/A:C	1	9	7.6	5.21%
AV:N/AC:L/Au:N/C:N/I:P/A:N	1	8	5	2.05%
AV:N/AC:L/Au:N/C:P/I:N/A:N	1	14	5	6.88%
AV:N/AC:M/Au:N/C:N/I:P/A:N	1	5	4.3	5.79%
AV:N/AC:M/Au:N/C:P/I:N/A:N	1	11	4.3	0.76%
AV:N/AC:M/Au:N/C:P/I:N/A:P	1	12	5.8	2.22%
Sum	95			

TABLE 13. Definition of vulnerability prioritization policies.

Policy	Description
FIFO	First-In-First-Out. First vulnerability in queue is first fixed.
CVSS	Vulnerability with the highest CVSS Score is first fixed.
VRSS	Vulnerability with the highest VRSS Score is first fixed.
HTAC	Vulnerability with the highest TAC value is first fixed.
FSTA	First highest CVSS Score - Then highest TAC value.
FATS	First highest TAC value - Then highest CVSS Score.

method may be equated with a random approach. This policy eliminates vulnerabilities in order they randomly appear in the IS. HTAC policy gives priority to vulnerabilities, which are exploitable by the highest number of threat agents (the highest TAC value). When two vulnerabilities have the same TAC value then the first found is first removed.

Two additional methods, which take into account the TAC value, were added to the list of policies. Namely, several CVSS vectors have exactly the same CVSS score. Therefore, it is reasonable in such case to give priority to a vulnerability having a higher TAC value. A policy that follows this rule is named FSTA. Similarly, several CVSS vectors have exactly the same TAC value. In such a case it makes sense to give priority to a vulnerability that has a higher CVSS score. Policy based on this rule is FATS.

Evaluation has to be carried out in an environment that guarantees exactly the same conditions for all compared policies. At the same time, the experimental environment must reflect the properties of a common IS. Since modern ISs consist of several various applications, the vulnerabilities of experimental information system can be represented by random vulnerabilities from NVD. The experimental environment was build as follows:

- An instance of IS was created with an empty list of vulnerabilities.
- 1,000 random vulnerabilities from NVD database were added to IS. Those with the existing PoC exploits in EDB are recognized as exploitable vulnerabilities or PoC vulnerabilities.

- For each policy from Table 13 one security guard was added to the system.
- At the start of simulation each security guard received its own clone of IS instance in order to assure exactly the same starting conditions.

Simulation model was implemented by way of applying the Recursive Porous Agent Simulation Toolkit (RePast) that is a free and open source agent-based modelling and simulation environment [28]. The effectiveness of policies was measured by $ExposureEV_k$ metric. Single simulation took 1,000 steps. In each step one vulnerability was fixed in each IS according to the policy priorities. After 1,000 steps all vulnerabilities were removed from all ISs. The main focus of evaluation in each simulation step was the number of remaining exploitable vulnerabilities in IS. Fig.1 shows the counting of remaining PoC vulnerabilities in a typical simulation process. In the presented case: at the start of the simulation, there were 56 PoC vulnerabilities out of 1,000. Some policies succeeded to remove all the PoC vulnerabilities before the end of simulation.

The effectiveness of policies ($ExposureEV_k$) was measured after each quarter of simulation: at 25% (step $k = 250$), 50% ($k = 500$), 75% ($k = 750$) and 100% ($k = 1000$). Simulation was run 1,000 times for each policy, which thus represents a large sample. Each member of sample (each run) is independent and identically distributed over the sample. Results are presented in Table 14 (mean values) and Table 15 (standard deviations). The pairwise comparisons of the mean values can be done by Z-test.

TABLE 14. Exposure to PoC vulnerabilities measured by $ExposureEV_k$ at the end of each quarter of simulation (mean values).

Policy	25%	50%	75%	100%
FATS	10,093	16,054	19,503	20,281
HTAC	10,090	16,251	19,714	20,545
FSTA	12,821	19,153	22,272	23,056
CVSS	12,824	19,177	22,348	23,080
VRSS	12,890	20,739	24,115	24,899
FIFO	12,579	21,567	26,973	28,749

TABLE 15. Exposure to PoC vulnerabilities measured by $ExposureEV_k$ at the end of each quarter of simulation (standard deviations).

Policy	25%	50%	75%	100%
FATS	1,468	2,541	3,265	3,471
HTAC	1,468	2,562	3,278	3,494
FSTA	1,745	2,751	3,377	3,577
CVSS	1,745	2,754	3,381	3,570
VRSS	1,754	2,939	3,570	3,761
FIFO	1,704	3,033	3,938	4,282

In the last evaluation step we observed how many PoC vulnerabilities were fixed in each quarter of simulation by each policy. Policies with better prediction abilities remove more PoC vulnerabilities in earlier quarters. Results are presented in Table 16.

TABLE 16. The number of removed PoC vulnerabilities by policy in each quarter (mean values).

Policy	25%	50%	75%	100%
	ALL			
CVSS	16.93	24.10	8.64	7.88
FATS	26.95	11.73	10.70	8.18
FIFO	14.56	14.20	14.45	14.35
FSTA	16.98	24.03	7.90	8.64
HTAC	26.72	11.81	10.84	8.19
VRSS	12.50	27.77	9.07	8.22
	CRITICAL			
CVSS	16.93	14.07	-	-
FATS	22.33	5.04	1.55	2.08
FIFO	7.86	7.58	7.77	7.79
FSTA	16.98	14.02	-	-
HTAC	22.28	4.98	1.63	2.11
VRSS	11.52	19.48	-	-
	MEDIUM			
CVSS	-	10.04	8.64	6.54
FATS	4.62	6.68	9.11	4.81
FIFO	6.37	6.29	6.33	6.24
FSTA	-	10.02	7.90	7.30
HTAC	4.43	6.83	9.17	4.79
VRSS	0.98	8.29	8.92	7.04
	LOW			
CVSS	-	-	-	1.33
FATS	-	-	0.04	1.29
FIFO	0.34	0.33	0.34	0.32
FSTA	-	-	-	1.33
HTAC	-	-	0.04	1.29
VRSS	-	-	0.15	1.19

V. DISCUSSION

Out of 21 threat agents from TAL Library, only 18 may exploit at least one CVSS vector (see Table 9). The highest TAC value is 18, yet only four vectors have this value (see Table 8). Another important observation is that less than 10% of vectors has a TAC value greater than 10. Moreover, most CVSS vectors have a TAC value equal to 0 (more than 20% of vectors).

However, when considering actual vulnerabilities (columns NVD, EDB and EKITS in Table 8), it may be found that a vast majority of vulnerabilities have a TAC value greater than zero. Hence, it may be concluded that vectors with a TAC value equal to zero rarely appear in real environments. 67% EDB and 96% EKITS vulnerabilities have TAC value greater than 10, however in NVD less than 50% of vulnerabilities have so high TAC value. The average TAC value is higher in EDB and EKITS databases, which represent exploitable vulnerabilities (see Table 17). We conclude that on average exploitable vulnerabilities have higher TAC values than non-exploitable.

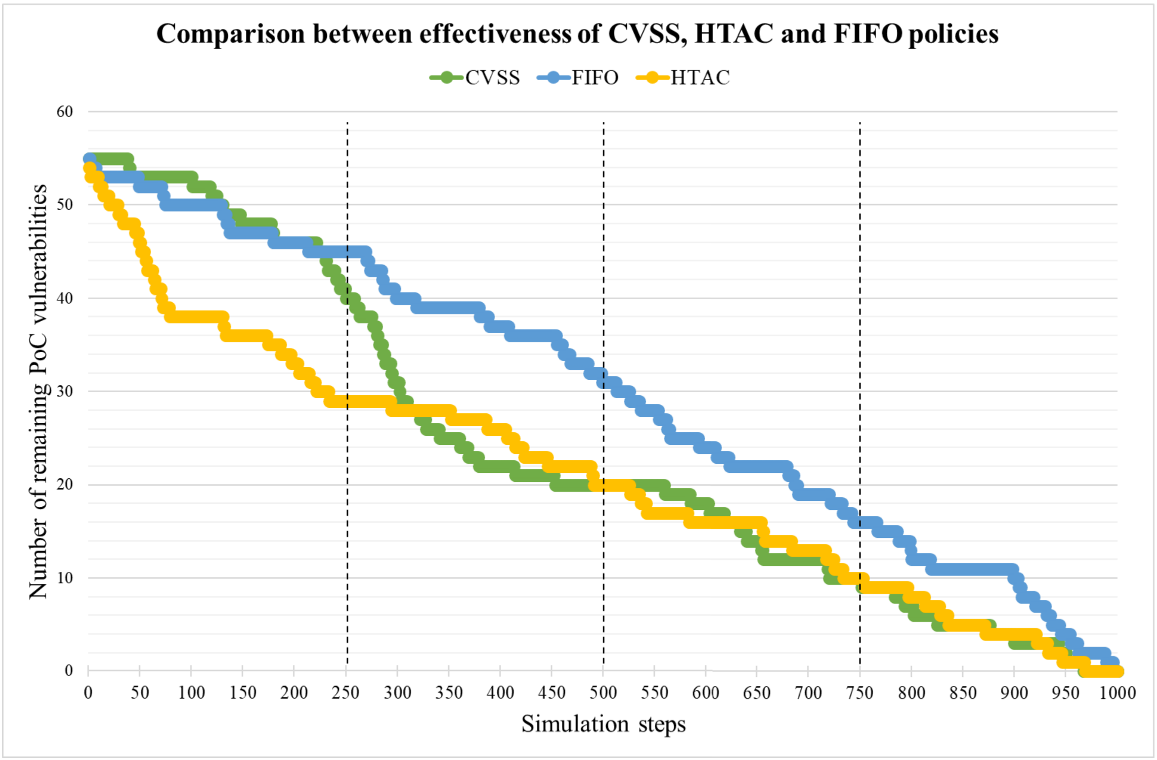

FIGURE 1. Results of a single simulation run. All policies start with the same set of vulnerabilities and in at most 1,000 steps remove all PoC vulnerabilities.

TABLE 17. Average, max and min TAC values of vulnerabilities in databases used in the experiment.

TAC	NVD	EDB	EKITS
Average	5.0	10.1	10.8
Max	18	18	18
Min	0	2	4

CVSS vector `[AV:N/AC:M/Au:N/C:N/I:P/A:N]` deserves special attention (the first row in Table 10). Its TAC value is 5, which is quite low. It is the most frequent CVSS vector among vulnerabilities in NVD, and 5.79% of them are also found in EDB (%PoC column). Summary data on these vulnerabilities in NVD prove that 3,607 out of 4,764 are XSS vulnerabilities, which is 80%. Interestingly, several authors claim that XSS vulnerabilities are assigned wrong CVSS vector. In addition to partially compromised integrity, XSS also allows partial compromising of confidentiality and availability [7], [14]. Such CVSS vector has a TAC value 13, which indicates that it is exploitable by much more threat agents.

A very common CVSS vector in NVD database is also `[AV:A/AC:M/Au:N/C:P/I:P/A:P]`. It is assigned to 1,423 vulnerabilities (eighth row in Table 10). Its TAC value is 13, which is quite high. However, in EDB database, a single vulnerability with this vector may not be found (0.0 value in %PoC column). A detailed overview of vulnerability summary in NVD reveals that the vast majority of these vulnerabilities (1,380 vulnerabilities amounting to 97%) have been disclosed in a relatively short time (in the last three months of 2014) and all relate to the same issue, "*library does not verify X.509 certificates from SSL servers, which allows man-in-the-middle attackers to spoof servers and obtain sensitive information via a crafted certificate*". The fact is that all these vulnerabilities have the same cause and that the response has been quick and massive. Otherwise, this CVSS vector is not so frequent in NVD database.

Results in Table 14 prove that the use of TAC value for vulnerability prioritization purposes is useful. HTAC policy assures 9% lower exposure to PoC vulnerabilities than CVSS. In the first quarter the method is even 20% more efficient than CVSS. FATS policy, which is similar to HTAC, yet takes into account also the CVSS score when TAC values are equal, achieved even a slightly better result. Statistically there are no differences between them. A similar situation may be observed when comparing CVSS and FSTA. VRSS method proves to be 6% worse than CVSS, while FIFO approach is 22% worse than CVSS (two sample Z-test for mean is used in all tests, $\alpha = 0.01$).

However, it is interesting that FIFO policy is slightly better than CVSS in the first quarter (see Table 14). The reason being that CVSS eliminates only vulnerabilities with the highest CVSS score in the first quarter, while FIFO removes vulnerabilities with arbitrary score. From Table 11 it is obvious that PoC vulnerabilities do not always have the highest CVSS scores.

The greater effectiveness of HTAC and FATS may be attributed to the fact that they have better prediction abilities. They remove higher number of PoC vulnerabilities in the first quarter of simulation than other policies (see Table 16). However, this is a very important feature, as there is the highest number of PoC vulnerabilities in IS at the very beginning, representing a higher risk. While the CVSS policy neutralizes vulnerabilities in the order from CRITICAL to LOW, HTAC policy neutralizes MEDIUM vulnerabilities in the first quarter as well. We concluded that TAC value is a good indicator of vulnerability exploitability. Higher the TAC value of vulnerability, PoC exploit for that vulnerability is more likely to exist.

Finally, the model certainly has some limitations. The simulated environment is a rough approximation of the situation in the wild. Attackers are a hidden population and we can only assume their characteristics (intents and behaviour). Furthermore, we have used the TAL library, which is just one possible description of the attackers. There are also other interesting characteristics that this library does not address. One such characteristic, which can be useful for measuring the effectiveness of prioritization policies, is the size of each group of threat agents.

VI. CONCLUSION AND FUTURE WORK

The paper focuses on proactive security, where it is crucial that the most severe vulnerabilities are to be removed first. The information system owner has to apply an effective prioritization policy to reduce risk. For this purposes the CVSS base score is commonly used, but several studies showed that it is not a good predictor of exploitable vulnerabilities. Alternative approaches with better prediction abilities are required. Furthermore, it is largely unexplored how effective is CVSS score compared to other methods. And this is where the main contribution of this research paper comes in.

Studies from security area suggest that incorporating attacker characteristics in risk estimation improves its accuracy. Therefore, the impact of attacker characteristics on vulnerability exploitation was examined. We have shown that the number of threat agents, which can exploit vulnerability, is a significantly better predictor of exploitable vulnerabilities comparing to CVSS score.

Moreover, the paper proposes a model for evaluating the effectiveness of vulnerability prioritization policies. For this purpose the paper defines a metric for measuring the exposure of IS to exploitable vulnerabilities. Our proposed HTAC method based on CVSS vectors and attacker characteristics, assures faster risk reduction than policy based only on a CVSS score.

In the future work, the model may expand in two directions. The first one is to take into account the size of threat agent groups in calculating TAC values. The other one is to include specific features of an information system that attract attackers into the model. Both extensions together represent a possibility for adapting the presented model to specific information system environments.

REFERENCES

[1] (2017). "Vulnerability review 2017." Secunia Res., Itasca, IL, USA, Tech. Rep. [Online]. Available: https://resources.flexera.com/web/pdf/Research-SVM-Vulnerability-Review-2017.pdf

[2] C. Fruhwirth and T. Männistö, "Improving CVSS-based vulnerability prioritization and response with context information," in *Proc. 3rd Int. Symp. Empirical Softw. Eng. Meas. (ESEM)*, 2009, pp. 535–544.

[3] M. Bozorgi, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond heuristics: Learning to classify vulnerabilities and predict exploits," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2010, pp. 105–114. [Online]. Available: http://dl.acm.org/citation.cfm?id=1835821

[4] P. Mell, K. Scarfone, and S. Romanosky, "Common vulnerability scoring system," *IEEE Security Privacy*, vol. 4, no. 6, pp. 85–89, Nov./Dec. 2006.

[5] H. Holm, M. Ekstedt, and D. Andersson, "Empirical analysis of system-level vulnerability metrics through actual attacks," *IEEE Trans. Dependable Secur. Comput.*, vol. 9, no. 6, pp. 825–837, Nov./Dec. 2012. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-84866600214&partnerID=tZOtx3y1

[6] L. Allodi and F. Massacci, "Comparing vulnerability severity and exploits using case-control studies," *ACM Trans. Inf. Syst. Secur.*, vol. 17, no. 1, pp. 1–20, 2014. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2660572.2630069

[7] H. Holm and K. Khan, "An expert-based investigation of the common vulnerability scoring system," *Comput. Secur.*, vol. 53, pp. 18–30, Sep. 2015. [Online]. Available: http://dx.doi.org/10.1016/j.cose.2015.04.012

[8] H. Ghani, J. Luna, and N. Suri, "Quantitative assessment of software vulnerabilities based on economic-driven security metrics," in *Proc. Int. Conf. Risks Security Internet Syst. (CRiSIS)*, Oct. 2013, pp. 1–8. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6766361

[9] R. Gandhi, A. Sharma, W. Mahoney, W. Sousan, Q. Zhu, and P. Laplante, "Dimensions of cyber-attacks: Social, political, economic, and cultural," *IEEE Technol. Soc. Mag.*, vol. 30, no. 1, pp. 28–38, Jan. 2011.

[10] L. Ben Othmane, R. Ranchal, R. Fernando, B. Bhargava, and E. Bodden, "Incorporating attacker capabilities in risk estimation and mitigation," *Comput. Secur.*, vol. 51, pp. 41–61, Jun. 2015. [Online]. Available: http://dx.doi.org/10.1016/j.cose.2015.03.001

[11] S. H. Houmb, V. N. L. Franqueira, and E. A. Engum, "Quantifying security risk level from CVSS estimates of frequency and impact," *J. Syst. Softw.*, vol. 83, no. 9, pp. 1622–1634, 2010. [Online]. Available: http://dx.doi.org/10.1016/j.jss.2009.08.023

[12] D. Trček, "Computationally supported quantitative risk management for information systems," in *Performance Models and Risk Management in Communications Systems*, vol. 46, N. Gülpınar, P. Harrison, and B. Rüstem, Eds. New York, NY, USA: Springer, 2011. [Online]. Available: https://doi.org/10.1007/978-1-4419-0534-5_3

[13] A. Chatzipoulidis, D. Michalopoulos, and I. Mavridis, "Information infrastructure risk prediction through platform vulnerability analysis," *J. Syst. Softw.*, vol. 106, pp. 28–41, Aug. 2015. [Online]. Available: http://dx.doi.org/10.1016/j.jss.2015.04.062

[14] Q. Liu and Y. Zhang, "VRSS: A new system for rating and scoring vulnerabilities," *Comput. Commun.*, vol. 34, no. 3, pp. 264–273, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.comcom.2010.04.006

[15] Q. Liu, Y. Zhang, Y. Kong, and Q. Wu, "Improving VRSS-based vulnerability prioritization using analytic hierarchy process," *J. Syst. Softw.*, vol. 85, no. 8, pp. 1699–1708, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.jss.2012.03.057

[16] S. Christey and B. Martin. (2013). *Buying Into the Bias: Why Vulnerability Statistics Suck*. [Online]. Available: https://www.blackhat.com/us-13/archives.html#Martin

[17] K. Nayak, D. Marino, P. Efstathopoulos, and T. Dumitraş, "Some vulnerabilities are different than others," in *Research in Attacks, Intrusions and Defenses (RAID)* (Lecture Notes in Computer Science), vol. 8688, A. Stavrou, H. Bos, and G. Portokalidis, Eds. Cham, Switzerland: Springer, 2014. [Online]. Available: https://doi.org/10.1007/978-3-319-11379-1_21

[18] S. Neuhaus, T. Zimmermann, C. Holler, and A. Zeller, "Predicting vulnerable software components," in *Proc. 14th ACM Conf. Comput. Commun. Security (CCS)*, 2007, pp. 529–540. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1315245.1315311

[19] V. H. Nguyen and L. M. S. Tran, "Predicting vulnerable software components with dependency graphs," in *Proc. 6th Int. Workshop Security Meas. Metrics-(MetriSec)*, 2010, pp. 1–8. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1853919.1853923

[20] S.-W. Woo, H. Joh, O. H. Alhazmi, and Y. K. Malaiya, "Modeling vulnerability discovery process in Apache and IIS HTTP servers," *Comput. Secur.*, vol. 30, no. 1, pp. 50–62, 2011. http://linkinghub.elsevier.com/retrieve/pii/S0167404810000908

[21] P. M. Mell, K. A. Scarfone, and S. Romanosky, "The common vulnerability scoring system (CVSS) and its applicability to federal agency systems," NIST, Gaithersburg, MD, USA, Tech. Rep. 7435, 2007. [Online]. Available: http://csrc.nist.gov/publications/nistir/ir7435/NISTIR-7435.pdf

[22] FIRST. (2015). *Common Vulnerability Scoring System V3.0: Specification Document*. [Online]. Available: https://www.first.org/cvss/cvss-v30-specification-v1.7.pdf

[23] A. Kott and C. Arnold, "The promises and challenges of continuous monitoring and risk scoring," *IEEE Security Privacy*, vol. 11, no. 1, pp. 90–93, Jan. 2013.

[24] S. K. Katsikas, "Risk management," in *Computer and Information Security Handbook*. Waltham, MA, USA: Elsevier, 2013, pp. 905–927. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780123943972000532

[25] C. P. Pfleeger and S. L. Pfleeger, *Security in Computing*, 4th ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2006.

[26] S. Vidalis and A. Jones, "Analyzing threat agents & their attributes," in *Proc. 5th Eur. Conf. Inf. Warfare Secur.*, 2005, pp. 1–15. [Online]. Available: https://www.researchgate.net/publication/220947230_Analyzing_Threat_Agents_and_Their_Attributes

[27] T. Casey. (2007). *Threat Agent Library Helps Identify Information Security Risks*. [Online]. Available: https://communities.intel.com/docs/DOC-1151

[28] M. J. North *et al.*, "Complex adaptive systems modeling with repast simphony," *Complex Adapt. Syst. Model.*, vol. 1, no. 1, pp. 1–26, 2013. [Online]. Available: http://link.springer.com/article/10.1186/2194-3206-1-3

ANDREJ DOBROVOLJC received the M.S. degree in computer science from the University of Ljubljana, Slovenia, where he is currently pursuing the Ph.D. degree in computer science. He has taken part in several application and research projects from computer science in various organizations. His primary research area is the information systems security. His research interests include business process modeling, business process analysis, and project management.

DENIS TRČEK received the Ph.D. degree from the University of Ljubljana, Slovenia, in 1995. He has been involved in the field of computer networks and IS security and privacy for over 20 years. He has taken part in various international and national research and application projects in government, banking, and insurance sectors. His interests include IS security, privacy, formal methods, trust management, and human factor modeling. His bibliography includes over one hundred titles, including two monographs published by Springer Nature. He serves as a member of many international bodies, boards, and evaluation committees.

BORUT LIKAR received the Ph.D. degree from the Faculty of Electrical Engineering, Ljubljana and the MBA degree from the IEDC-Bled School of Management. He has authored many scientific and expert publications. He is an innovator, initiator, and the head of several successful international R&D projects and the author of many patents, models, and copyright works, of which many proved to be extremely marketable. He is the Founder of the National Association of Innovative Youth. He is an Evaluator of EU projects and an invited speaker.