Comparing vulnerability severity and exploits using case-control studies.

LUCA ALLODI, University of Trento
FABIO MASSACCI, University of Trento

(U.S) Rule-based policies to mitigate software risk suggest using the CVSS score to measure the risk of an individual vulnerability and "act" accordingly. A key issue is whether the "danger" score does actually match the risk of exploitation in the wild, and if and how such score can be improved.

To address this question we propose to use a case-control study methodology similar to the procedure used to link lung cancer and smoking in the 1950s. A case-control study allows the researcher to draw conclusions on the relation between some *risk factor* (e.g. smoking) and an effect (e.g. cancer) by looking backward at the *cases* (e.g. patients) and comparing them with *controls* (e.g. randomly selected patients with similar characteristics). The methodology allows to quantify (statistically) the *risk reduction* achievable by acting on the risk factor. We illustrate the methodology by using publicly available data on vulnerabilities, exploits and exploits in the wild to (1) evaluate the performances of the current risk factor in the industry, the CVSS base score; (2) determine whether it can be improved by considering additional factors such the existence of a proof-of-concept exploit, or of an exploit in the black markets. Our analysis reveals that (a) fixing a vulnerability just because it was assigned a high CVSS score is equivalent to randomly picking vulnerabilities to fix; (b) the existence of proof of concept exploits is a significantly better risk factor; (c) fixing in response to exploit presence in black markets yields the largest risk reduction.

Categories and Subject Descriptors: D.2.9 [**Software Engineering**]: Management

General Terms: Security, Management, Measurement

Additional Key Words and Phrases: Software vulnerability, exploitation, CVSS, patching, compliance

ACM Reference Format:
Luca Allodi and Fabio Massacci, 2013. Comparing vulnerability severity and exploits using case-control studies. *ACM Trans. Embedd. Comput. Syst.* 9, 4, Article 39 (March 2013), 21 pages.
DOI:http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

Software security configuration managers (e.g. Tripwire Enterprise, HP SCAP Scanner, QualysGuard FDCC Module, Rapid 7 Nexpose) usually rely on the National (US) Vulnerability Database[1] (NVD for short). Each vulnerability is reported alongside a

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

This work was partly supported by the EU-SEC-CP-SECONOMICS and MIUR-PRIN-TENACE Projects. We would like to thank Tudor Dimitras and Viet H. Nguyen for the many useful discussions. Viet actually wrote the second script used for cross-checking the SYM dataset. We also thank Julian Williams for his very useful feedback. Further thanks go to the anonymous TISSEC reviewers that greatly helped us in making this paper a better one. No statement in this paper should be interpreted as an endorsement by Symantec.
Author's addresses: Luca Allodi and Fabio Massaci, Department of Information Engineering and Computer Science (DISI), University of Trento, Italy.

DOI:http://dx.doi.org/10.1145/0000000.0000000

'technical assessment' given by the Common Vulnerability Scoring System[2] (CVSS), which evaluates different technical aspects of the vulnerability [Mell et al. 2007].

The CVSS score is often used as a metric for risk, despite it not being designed for this purpose. For example, the US Federal government with QTA0-08-HC-B-0003 reference notice requires all IT products for the US Government to manage and assess the security of IT configurations with the NIST certified S-CAP protocol [Quinn et al. 2010], which explicitly says: "*Organizations should use CVSS base scores to assist in prioritizing the remediation of known security-related software flaws based on the relative severity of the flaws.*". Another notable example is PCI DSS, the standard for security of credit card data, that states a similar rule: "*Risk rankings should be based on industry best practices. For example, criteria for ranking High risk vulnerabilities may include a CVSS base score of 4.0 or above [..].*" [Council 2010]. As a result, the CVSS base score is now commonly used in industry to identify 'high risk' vulnerabilities that must be fixed with the highest priority. As of the date of publication it is not clear to what degree the CVSS score correlates with attacks in the wild. Acknowledging the problem, different tools in the industry (e.g. Rapid7's, Qualy's, Symantec's and Tripwire's) consider risk factors other than the sole CVSS when evaluating a system's security. However, a sound comparison between different security policies is hard to provide in this context as the security community still lacks of a *scientific methodology* to compare the effectiveness of policies that account for different risk factors.

A major obstacle in having a clear understanding on this is the nature of the data at hand. Vulnerability-database information is rife with problems and its use to classify risks of exploit is often inappropriate (see e.g [Frei et al. 2006; Shahzad et al. 2012; Houmb et al. 2010] as some examples). For example, a common problem is to use proof-of-concept exploit data to measure software security or vendor performances. While proof-of-concept exploit data is much easier to collect than data on actual attacks, the former says little about the workability of the exploit and the state of security of the vulnerable software: on the contrary, a proof-of-concept exploit is merely a byproduct of the so-called *responsible vulnerability disclosure* process, whereby a security researcher that finds a vulnerability discloses it to the vendor alongside a proof-of-concept exploitation code that proves the existence of the vulnerability itself [Miller 2007].

Software and vulnerability risk measures should be based on factual evidence of exploitation rather than on security researchers's participation in bug bounty programs. For example, in [Shahzad et al. 2012] the authors compare the security of different vendors' products by comparing CVSS scores and "zero-day" exploits[3] by subtracting two dates in a public database of proof-of-concept exploits. Given the nature of the analyzed data, conclusions on the security of different vendors and their respective products may be misleading. Software vulnerability data is often unreliable as well; for example, vulnerability timing data in public databases such as the National Vulnerability Database may "*contain errors of unknown size*" [Schryen 2009]. This is again due to the nature of the disclosure mechanism. In conclusion, a major problem when assessing software security and vulnerability exploits is to identify workable data on vulnerabilities and vulnerability exploitation (whether attempted or successful) and handle it properly by means of a methodology that accounts for the data's inherent limitations.

To address these problems, in this paper we:

[2] http://www.first.org/cvss

[3] A zero-day exploit is present when the exploit is reported before or on the date that the vulnerability is disclosed.

(1) Present our datasets of vulnerabilities, proof-of-concept exploits, exploits traded in the black markets and exploits detected in the wild.
(2) Introduce the case-control study as a fully-replicable methodology to soundly analyze vulnerability and exploit data.
(3) Check the suitability of the current use of the CVSS score as a risk metric by comparing it against *actual exploits recorded in the wild* and by performing a breakdown analysis of its characteristics and values.
(4) We use the case-control study methodology to show how one can improve the current practice of "Base CVSS" by considering other risk factors and quantitatively assess their performance in terms of risk reduction. The risk factors considered in our study are:
    (a) The CVSS base score as reported by the National Vulnerability Database.
    (b) Existence of a public proof-of-concept exploit.
    (c) Existence of an exploit traded in the cybercrime black markets.
    Any other risk factors, like software popularity, CVSS subscores, or other measurable values may be considered when replicating our methodology.

An important facet of our methodology is its reproducibility and extensibility to many practical scenarios. In order to illustrate these advantageous properties we a) provide an exhaustive description of the analytical procedure and the rationale behind the specific decisions needed to operationalise it; b) make our datasets available for replication and robustness checks beyond the scope of this paper.

The remainder of this paper is organised as follows. We first introduce our four datasets (§2), illustrate the problem with the current CVSS-based best practice (§2.1) and provide a breakdown of the issue (§3). In the core of the paper we propose a methodology to precisely assess the performances of the CVSS score and other risk factors (§4). We then discuss our results (§5) and this study's threats to validity (§6). We finally review related work (§7) and conclude (§8).

2. DATASETS

We base our analysis on four datasets:

— NVD (National Vulnerability Database): the "universe" of vulnerabilities. NVD is the reference database for disclosed vulnerabilities held by NIST. It has been widely used and analyzed in previous vulnerability studies [Massacci et al. 2011; Shahzad et al. 2012; Scarfone and Mell 2009]. Our copy of the NVD dataset contains data on 49599 vulnerabilities reported until June 2012.
— EDB (Exploit-db[4]): proof-of-concept exploits. EDB includes information on proof-of-concept exploits and references the respective CVE. Our EDB copy contains data on 8122 *proof-of-concept* exploits and affected CVEs.
— EKITS: black-marketed exploits. EKITS is our dataset of vulnerabilities bundled in Exploit Kits. Exploit Kits are malicious web sites that the attacker deploys on some public webserver he/she controls. Their purpose is to attack and infect systems that connect to them. For further details refer to [Kotov and Massacci 2013; Grier et al. 2012]. EKITS is based on Contagio's Exploit Pack Table[5] and, at the time of writing, represents a substantial expansion over it in terms of reported exploit kits. EKITS reports exploits for 103 unique CVEs bundled in 90+ exploit kits. A sample of notable names of those are: *Elenonore, Blackhole, Crimepack, Fragus, Sakura, Icepack*.

[4] http://www.exploit-db.com/
[5] http://contagiodump.blogspot.it/2010/06/overview-of-exploit-packs-update.html

Table I. Summary of our datasets

DB	Content	Collection method	#Entries
NVD	CVEs	XML parsing	49599
EDB	Publicly exploited CVEs	Download and web parsing to correlate with CVEs	8122
SYM	CVEs exploited in the wild	Web parsing to correlate with CVEs	1277
EKITS	CVEs in the black market	ad-hoc analysis + Contagio's Exploit table	103

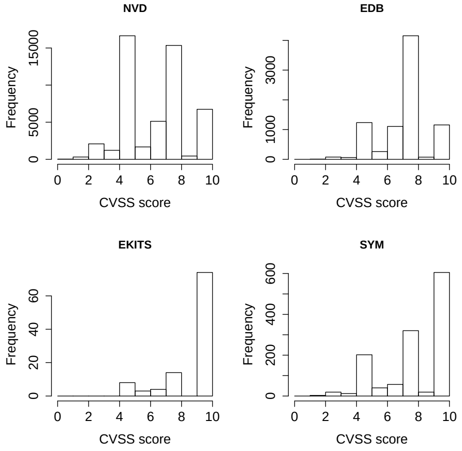


Fig. 1. Distribution of CVSS scores per dataset.

— SYM: vulnerabilities exploited in the wild. SYM reports vulnerabilities that have been exploited in the wild as documented in Symantec's AttackSignature[6] and ThreatExplorer[7] public datasets. SYM contains 1277 CVEs identified in viruses (local threats) and remote attacks (network threats) by Symantec's commercial products. This has of course some limitation as direct attacks by individual motivated hackers against specific companies are not considered in this metric. The SYM dataset can be seen as an "index" of the wider WINE dataset [Dumitras and Shou 2011] where actual volumes of attacks are reported. We do not use it here as we are trying to characterize a worst case scenario where "one exploit is too many".

Table I summarizes the content of each dataset and their respective collection methodologies. For further details see [Allodi and Massacci 2012], all of the datasets used in this study are available from the authors on request[8].

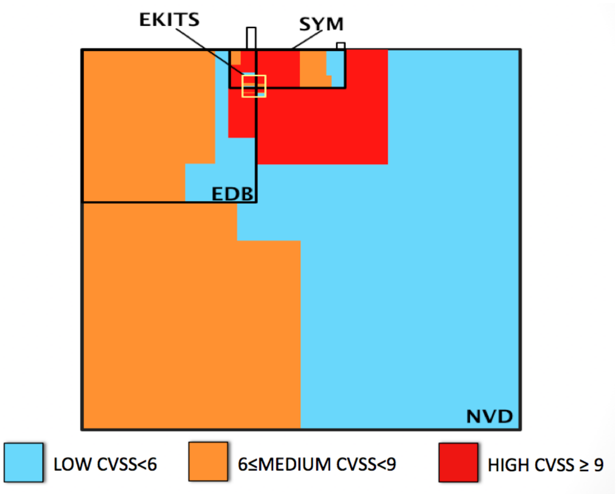
2.1. A coarse-grained overview of the datasets

The CVSS score is represented by a number in $[0..10]$, where 0 is the lowest criticality level and 10 the maximum (for further reference see [Mell et al. 2007]). We report in Figure 1 the histogram distribution of the CVSS base scores. Three clusters of vulnerabilities are visually identifiable throughout our datasets:

[6] http://www.symantec.com/security_response/attacksignatures/

[7] http://www.symantec.com/security_response/threatexplorer/

[8] http://securitylab.disi.unitn.it/doku.php?id=datasets



Dimensions are proportional to data size. In red vulnerabilities with CVSS$\geq$9 score. Medium score vulnerabilities are orange, and cyan represents vulnerability with CVSS lower than 6. The two small rectangles outside of NVDspace are vulnerabilities whose CVEs were not present in NVD at the time of sampling.

Fig. 2. Relative Map of vulnerabilities per dataset

(1) HIGH: CVSS ≥ 9
(2) MEDIUM: $6 \leq$ CVSS < 9
(3) LOW: CVSS < 6

The role of the CVSS score is, in the context of our analysis, to discern dangerous vulnerabilities from non-dangerous ones. Therefore an important analysis at this stage is to understand the overlap between the datasets, in order to grasp whether they and the CVSS score are capturing the same phenomenon.

In Figure 2 we report a Venn diagram of our data. Area size is proportional to the number of vulnerabilities that belong to it; the color is an indication of the CVSS score. Red, orange and cyan areas represent HIGH, MEDIUM and LOW score vulnerabilities respectively. This map gives a first intuition of the problem with the CVSS base score as a risk metric for exploitation: the "red area" located *outside* of SYM corresponds to CVSS false positives (i.e. HIGH "risk" vulnerabilities that are not exploited), while *within* SYM about half the vulnerabilities are false negatives (i.e. LOW and MEDIUM "risk" vulnerabilities that *are* exploited). In other words the CVSS score misses a high rate of exploited vulnerabilities and erroneously marks as HIGH risk a high rate of non-exploited vulnerabilities.

Table II reports the likelihood of a vulnerability being in SYM if it is contained in one of our datasets. Prima facie analysis would suggest that there is approximately a 75% probability that a vulnerability in the black markets is exploited in the wild. For NVD and EDB the rate of exploited vulnerabilities is less than 5%. However, these conclusions can be *grossly* incorrect. For example SYM might report only vulnerabilities of interest to Symantec's costumers. Suppose they mostly use Windows; then all Linux vulnerabilities listed in EDB would not be mentioned in SYM not because they are not exploited in the wild, but simply because they are not interesting for Symantec. Another possible example can be that Symantec mainly detects "remote code execution" vulnerabilities, while NVD might report lots of vulnerabilities exploitable through social engineering. We might therefore have a "selection bias" problem. In order to offer more scientifically sound conclusions we need to (a) better understand the internals of

Table II. A (potentially erroneous) conditional probability of vulnerability being a threat

	vuln in SYM	vuln not in SYM
EKITS	75.73%	24.27%
EDB	4.81%	95.19%
NVD	2.57%	97.43%

Note: Conditional probability that a vulnerability v is listed by Symantec as threat knowing that it is contained in a dataset, i.e. $P(v \in SYM \mid v \in dataset)$. This is a rushing computation because datasets might be constructed with different criteria.

Table III. Possible values for the Exploitability and Impact subscores.

Impact subscore			Exploitability subscore		
Confidentiality	Integrity	Availability	Access Vector	Access complexity	Authentication
Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
None	None	None	Local	High	Multiple
Partial	Partial	Partial	Adjacent Net.	Medium	Single
Complete	Complete	Complete	Network	Low	None

the CVSS base score (which we do in the next subsection) and (b) propose a methodology to make sure we are comparing apples with apples (which we do in the case-control methodology section, §4).

3. CVSS SCORE BREAKDOWN

The Common Vulnerability Scoring System identifies three scores: the *base score*, the *temporal score*, and the *environmental score* [Mell et al. 2007]. The base score identifies "fundamental characteristics of a vulnerability that are constant over time and user environments"; the temporal score considers assessments like existence of a patch for the vulnerability, or the presence of an exploit in the wild; the environmental score considers further assessments tailored around the particular system implementation in which the vulnerability is present. However, of the three only the *base score* is identified, by standards and best practices alike, as the metric to rely upon for vulnerability management [Williams and Chuvakin 2012; Quinn et al. 2010]. The base score is also the only one commonly available in vulnerability bulletins and public datasets. We therefore only consider the base score in our analysis.

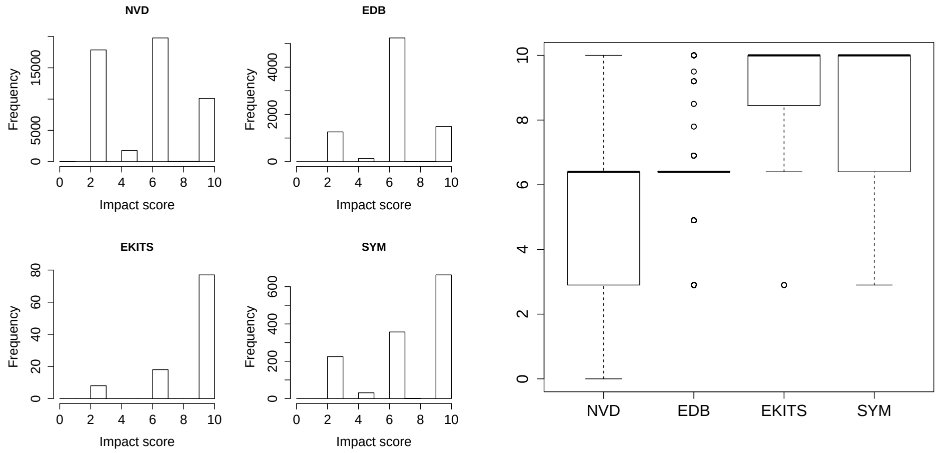
The CVSS base score is computed as a product of two submetrics: the Impact submetric and the Exploitability submetric. Therefore, the CVSS base score $CVSS_b$ is of the following form:

$$CVSS_b = Impact \times Exploitability \tag{1}$$

which closely recalls the traditional definition of risk as "impact × likelihood". The Impact submetric is an assessment of the impact the exploitation of the vulnerability has on the system. The Exploitability subscore is defined by factors such as the difficulty of the exploitation and reachability of the vulnerability (e.g. from the network or local access only). For this reason it is sometimes interpreted as a measure of "likelihood of exploit" (e.g. in [Bozorgi et al. 2010]).

3.1. The Impact and Exploitability Subscores

The Impact and Exploitability subscores are calculated on the basis of additional variables, reported in Table III. The Impact submetric is identified by three separate assessments on the Confidentiality, Integrity and Availability impacts on a victim sys-



The histogram on the left represents the frequency distribution of CVSS Impact values among the datasets. The boxplot on the right reports the distribution of values around the median (represented by a thick horizontal line). Outliers are represented by dots.

Fig. 3. Histogram and boxplot of CVSS Impact subscores per dataset.

tem. In this manuscript this triplet is referred to as the "CIA" impact. Each variable can assume three values: Complete (C), Partial (P), None (N).

The Exploitability submetric is as well identified by three variables:

—*Access Vector* gives information on the accessibility of the vulnerability by distinguishing the case when the attacker can exploit it remotely from the Network, (N); from an Adjacent Network (A); Locally (L).
—*Access Complexity* provides information on the difficulty the attacker may encounter in recreating the conditions for the vulnerability to be exploited. This assessment can assume three values: High (H), Medium (M), or Low (L).
—*Authentication* represents the number of steps of authentication the attacker has to go through to trigger the vulnerability. The levels of the assessment can be: None (N), Single (S), Multiple (M).

Table III reports a summary of the CVSS base score's variables and respective possible values.

3.2. Breakdown of the Impact subscore

Figure 3 depicts a histogram distribution of the Impact subscore. From inspection of the vulnerabilities it is apparent that the subscore does not assume all of the possible values. and as a result This is clearly visible from the subscore distribution, which has gaps below score 2, between 3 and 6 and between 7 and 9. Still, the Impact subscore reveals some variability throughout all datasets. In EDB scores between 6 and 7 characterize the great majority of vulnerabilities. This distribution may be an effect of the nature of the dataset: EDB features proof-of-concepts for vulnerabilities discovered by security researchers, likely with the intent of selling them to the software vendors; lower score vulnerabilities may be of too little value to be worth the bounty [Miller 2007]; medium-score ones may instead represent the "low-hanging fruits" that maximize the researchers' return-on-investment.

The great majority of vulnerabilities in SYM and EKITS have a Impact subscore greater than 6; unsurprisingly, vulnerabilities exploited in real attacks in the wild

Table IV. Incidence of values of CIA triad within NVD.

Confidentiality	Integrity	Availability	Absolute no.	Incidence
C	C	C	9972	20%
C	C	P	0	-
C	C	N	43	<1%
C	P	C	2	<1%
C	P	P	13	<1%
C	P	N	3	<1%
C	N	C	15	<1%
C	N	P	2	<1%
C	N	N	417	1%
P	C	C	5	<1%
P	C	P	1	<1%
P	C	N	0	-
P	P	C	22	-
P	P	P	17550	35%
P	P	N	1196	2%
P	N	C	9	<1%
P	N	P	110	<1%
P	N	N	5147	10%
N	C	C	64	<1%
N	C	P	1	<1%
N	C	N	43	<1%
N	P	C	17	<1%
N	P	P	465	1%
N	P	N	7714	16%
N	N	C	1769	4%
N	N	P	5003	10%
N	N	N	16	<1%

tend to yield a higher Impact on the victim system than the average vulnerability or proof-of-concept exploit.

The different distribution of the CVSS Impact subscore among the datasets is apparent in the boxplot reported in Figure 3. NVD results distributed in the whole range [0..10], with median just above 6 (6.4). The distribution of impact values in EDB is highly dense around the median (6.4). The distribution of the Impact subscore for SYM and EKITS are clearly different from the other two datasets; their median impact score of 10 is also significantly higher than those of NVD and EDB.

To explain the gaps in the histogram in Figure 3, we decompose the distribution of Impact subscores throughout our datasets. To simplify discussion, in Table IV we report the incidence of the existing values for the CIA assessments in NVD only. It is immediate to see that only few values are actually used. For example there is only one vulnerability whose CIA impact is “PCP” (i.e. partial impact on confidentiality, complete on integrity and partial on availability). Availability almost always assumes the same value of Integrity, apart from the case where there is no impact on Confidentiality, and looks therefore of limited importance for a descriptive discussion.

For the sake of readability, we therefore exclude the Availability assessment from the analysis, and proceed by looking at the two remaining Impact variables in the four datasets. This analysis is reported in Table V. Even with this aggregation on place many possible values of the CIA assessment result unused. “PP” vulnerabilities characterize the majority of disclosed vulnerabilities (NVD) and vulnerabilities with a proof-of-concept exploit (EDB). This observation changes completely when looking at the SYM and EKITS datasets, for which most vulnerabilities (50%, 75%) score “CC”. This shift alone can be considered responsible for the different distribution of scores depicted in Figure 3.

Table V. Combinations of Confidentiality and Integrity values per dataset.

Confidentiality	Integrity	SYM	EKITS	EDB	NVD
C	C	51.61%	74.76%	18.11%	20.20%
C	P	0.00%	0.00%	0.02%	0.04%
C	N	0.31%	0.97%	0.71%	0.87%
P	C	0.00%	0.00%	0.01%	0.01%
P	P	27.80%	16.50%	63.52%	37.83%
P	N	7.83%	0.97%	5.61%	10.62%
N	C	0.23%	0.00%	0.18%	0.22%
N	P	4.39%	2.91%	5.07%	16.52%
N	N	7.83%	3.88%	6.75%	13.69%

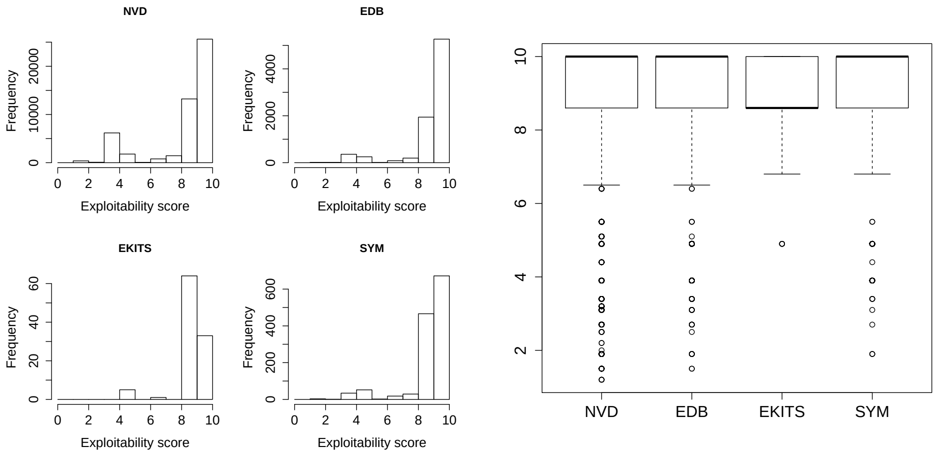


Fig. 4. Distribution of CVSS Exploitability subscores.

Table VI. Exploitability Subfactors for each dataset.

	metric	value	SYM	EKITS	EDB	NVD
Exploitability	Acc. Vec.	local	2.98%	0%	4.57%	13.07%
		adj.	0.23%	0%	0.12%	0.35%
		net	96.79%	100%	95.31%	86.58%
	Acc. Com.	high	4.23%	4.85%	3.37%	4.70%
		medium	38.53%	63.11%	25.49%	30.17%
		low	57.24%	32.04%	71.14%	65.13%
	Auth.	multiple	0%	0%	0.02%	0.05%
		single	3.92%	0.97%	3.71%	5.30%
		none	96.08%	99.03%	96.27%	94.65%

3.3. Breakdown of the Exploitability subscore

Figure 4 reveals the distribution of the Exploitability subscore for each dataset. Almost all vulnerabilities score between 8 and 10, and from the boxplot it is evident that the distribution of exploitability subscores is indistinguishable among the datasets. In other words, Exploitability can not be used as a proxy for likelihood of exploitation in the wild. A similar result but only for proof-of-concept exploits has also been reported in [Bozorgi et al. 2010]).

In table VI we decompose the Exploitability subscores and find that most vulnerabilities in NVD do not require any authentication (Authentication = (N)one, 95%), and are accessible from remote (Access Vector = (N)etwork, 87%). This observation is even more extreme in datasets other than NVD.

For this reason the CVSS Exploitability subscore resembles more a constant than a variable, and has therefore little or no influence on the variance of the final CVSS score. This may in turn affect the suitability of the CVSS as a *risk* metric, that would lack of a characterization of "exploitation likelihood".

4. RANDOMIZED CASE-CONTROL STUDY

Randomized Block Design Experiments (or Controlled Experiments) are common frameworks used to measure the effectiveness of a treatment over a sample of subjects. These designs aim at measuring a certain variable of interest by isolating factors that may influence the outcome of the experiment, and leave to randomization other factors of not primary importance. However, in some cases practical and ethical concerns may make an experiment impossible to perform; for example, one cannot ask subjects to start smoking in order to see whether they die of cancer. Similarly, we can not ask subjects to stay vulnerable to see if they get their computers infected and their bank accounts emptied.

When an experiment is not applicable, an alternative solution is to perform a retrospective analysis in which the *cases* (people with a known illness) are compared with a random population of *controls* clustered in "blocks" (randomly selected patients with the same characteristics). These retrospective analyses are called *Randomized case-control studies* and are in many respects analogous to their experimental counterpart. A famous application of this methodology is the 1950 study by [Doll and Hill 1950], where the authors showed the correlation between smoking habits and the presence or absence of cancer of the lungs by performing a case-control study with data on hospitalization.

We revisit this methodology to assess whether a vulnerability risk factor (like the CVSS score) can be a good predictor for vulnerability exploitation, and whether it can be improved by additional information.

We start by giving the reader some terminology:

— *Cases*. The cases of a control study are the subjects that present the observed effect. For example, in the medical domain the cases could be the "patients" whose status has been ascertain to be "sick". In a computer security scenario, a "case" could be a vulnerability that has been exploited in the wild. For us a case is therefore a vulnerability in SYM.

— *Explanatory variable or risk factor*. A risk factor is an effect that can explain the presence (or increase in likelihood) of the illness (or attack). For cancer it is smoking habits. We consider as risk factors (1) the CVSS level; (2) the existence of a Proof-of-Concept exploit ($vuln \in \mathsf{E}DB$); (3) the presence of an exploit in the black markets ($vuln \in \mathsf{E}KITS$). Another possibility (not investigated here) could be to use some of the CVSS subscores.

— *Confounding variables* can be other variables that may be alternative explanations of the effect, or correlate with its observation. For example, patient age or sex may be confounding factors for some types of cancer. In our case the existence of an exploit in SYM may depend on factors such as the type of vulnerability, its time of disclosure and the affected software (see the Linux vs Windows example in Section 2).

— *Control group*. A control group is a group of subjects chosen at random from a population with similar characteristics (e.g. age, social status, location) to the cases. In the original formulation of case-control study, the control group was composed of healthy people only. With that application of the case-control study we can only ascertain whether the observed effect (e.g. cancer of the lung) is related to a particular risk factor (e.g. smoking habits) by a greater or lower degree than to other confounding variables (e.g. living in polluted cities). We relax this condition and leave open the (random) chance that cases get included in the control group. This relaxation allows us to perform additional computations on our samples (namely CVSS sensitivity, specificity and risk reduction). This, however, introduces (random) noise in the generated data. To address this issue, we perform the analysis with bootstrapping.

- *Bootstrapping* is a technique by which noise in the data is "flattened" by re-sampling the data multiple times with replacement. This mitigates the effects, in the final analysis, of a random observation showing up in an iteration.

Confounding variables. Deciding which confounding factors to include in a case-control study is usually left to the intuition and experience of the researcher [Doll and Hill 1950]. Because SYM is the "critical point" of our study (as it reports our cancer patients), we consulted with Symantec to decide which factors to consider as confounding. While this list can not be considered an exhaustive one, we believe the variables we identify in the following capture the most important aspects of the inclusion of the vulnerability in SYM. More details on this process are discussed in the Threats to Validity Section (§6). In the following we discuss the confounding variables we choose and the enforcement of the respective controlling procedure:

- **Year.** Symantec's commitment in reporting exploited CVEs may change with time. After a detailed conversation with Symantec emerged that the inclusion of a CVE in an attack signature is an effort on Symantec's side aimed at enhancing the usefulness of their datasets. Specifically, Symantec recently opened a data sharing program called WINE whose aim is to share attack data with security researchers [Dumitras and Shou 2011]. The data included in the WINE dataset spans from 2009 to the present date. Given the explicit sharing nature of their WINE program, we consider vulnerabilities disclosed after 2009 to be better represented in SYM. We therefore consider only those in our study.
  Enforcement: Unfortunately vulnerability time data in NVD is very noisy due to how the vulnerability disclosure mechanism works [Schryen 2009; Miller 2007]. For this reason, an *exact match* for the disclosure date of the sampled vulnerability sv_i and the SYM vulnerability v_i is undesirable. In our case a coarse time data granularity is enough, as we only need to cover the years in which Symantec actively reported attacked CVEs. We therefore enforce this control by first selecting for sampling only vulnerabilities whose disclosure dates span from 2009 on, and then by performing an exact match in the year of disclosure between sv_i and v_i.
- **Impact type.** Our analysis (Section 3.2) showed that some CIA types are more common in SYM than elsewhere (e.g. CIA="CCC"). An explanation for this may be that attackers contrasted by Symantec may prefer to attack vulnerabilities that allow them to execute arbitrary code rather than ones that enables them to get only a partial access on the file system. We therefore also control for the CVSS Confidentiality, Integrity and Availability assessments.
  Enforcement: The CVSS framework provides a precise assessments of the CIA impact. We therefore perform an exact match between the CIA assessment of sv_i and that of v_i.

In addition, we "sanitize" the data by *Software.* Symantec is a security market leader and provides a variety of security solutions but its largest market share is in the consumer market. In particular, the data in SYM is referenced to the malware and attack signatures included in commercial products that are often installed on "consumer" machines. These are typically Microsoft Windows machines running commodity software like Microsoft Office and internet plugins like Adobe Flash or Oracle Java [Dumitras

Table VII. Output format of our experiment.

Risk Factor level	$v \in \mathsf{S}YM$	$v \notin \mathsf{S}YM$
Above Threshold	a	b
Below Threshold	c	d

Table VIII. Sample thresholds

CVSS $\geq$ 6
CVSS $\geq$ 9
CVSS $\geq$ 9 & v $\in$ EDB
CVSS $\geq$ 9 & v $\in$ EKITS

and Efstathopoulos 2012].[9] Because of this selection problem, SYM may represent only a subset of all the software reported in NVD or EDB or EKITS.

Enforcement: Unfortunately no standardized way to report vulnerability software names in NVD exists, and this makes it impossible to use this variable as a direct control. For example, CVE-2009-0559 (in SYM) is reported in NVD as a "*Stack-based buffer overflow in Excel*", but the main affected software reported is (Microsoft) Office. In contrast, CVE-2010-1248 (in SYM as well) is a "*Buffer overflow in Microsoft Office Excel*" and is reported as an Excel vulnerability. Thus, performing a perfect string match for the software variable would exclude from the selection relevant vulnerabilities affecting the same software but reporting different software names.

The problem with software names extends beyond this. Consider for example a vulnerability in *Webkit*, an HTML engine used in many browsers (e.g. Safari, Chrome, and Opera). Because Webkit is a component of other software, a vulnerability in Apple Safari might also be a Webkit vulnerability in Google Chrome.

For these reasons using string matching for "software" when selecting sv_i would introduce unknown error in the data. We can therefore only perform a "best effort" approach by checking that the software affected by sv_i is included in the list of software for $\forall v_i \in \mathsf{S}YM$. In this work *software* is therefore used as a "sanitation" variable rather than a proper control. A possible refinement of this to be considered for future work is to cluster software names in more general categories, e.g. "Browser" or "Plugin".

4.1. Experiment run

We divide our experiment in two parts: sampling and execution. In the former we generate the samples from NVD, EDB and EKITS. In the latter we compute the relevant statistics on the samples. What follows is a textual description of these processes. Our R script to replicate the data analysis is available on our Lab's webpage[10].

Sampling. To create the samples, we first select a vulnerability v_i from SYM and set the controls according to their values for v_i. Then, for each of NVD, EDB and EKITS we randomly select, with replacement, a sample vulnerability sv_i that satisfies the conditions defined by v_i. We then include sv_i in the list of selected vulnerabilities for that dataset sample. We repeat this procedure for all vulnerabilities in SYM. The sampling has been performed with the statistical tool R-CRAN [R Core Team 2012].

Execution. Once we collected our samples, we compute the frequency with which each risk factor identifies a vulnerability in SYM. Our output is represented in Table VII. Each risk factor is defined by a CVSS threshold level t in combination with the existence of a PoC ($v \in \mathsf{E}DB$) or of a black-marketed exploit ($v \in \mathsf{E}KITS$). Examples of thresholds for different risk factors are reported in Table VIII. We run our experiment for all CVSS thresholds t_i with $i \in [1..10]$. For each risk factor we evaluate the number of vulnerabilities in the sample that fall *above* and *below* the CVSS threshold, and that are included (or not included) in SYM: the obtained table reports the count of

[9] Unix software is also included in SYM. However we do not consider this sample to be representative of Unix exploited vulnerabilities.

[10] https://securitylab.disi.unitn.it/doku.php?id=software

vulnerabilities that each risk factor correctly and incorrectly identifies as “at high risk of exploit” ($\in$ SYM) or “at low risk of exploit” ($\not\in$ SYM).

The computed values depend on the random sampling process. In an extreme case we may therefore end up, just by chance, with a sample containing only vulnerabilities in SYM and below the current threshold (i.e. $[a = 0; b = 0; c = 1277; d = 0]$). Such an effect would be likely due to chance alone. To mitigate this we repeat, for every risk factor, the whole experiment run 400 times and keep the median of the results. We choose this limit because we observed that around 300 repetitions the distribution of results is already markedly Gaussian. Any statistic reported in this paper is to be intended as the median of the generated distribution of values.

4.2. Parameters of the analysis

Sensitivity and specificity. In the medical domain, the sensitivity of a test is the conditional probability of the test giving positive results when the illness is present. The specificity of the test is the conditional probability of the test giving negative result when there is no illness. Sensitivity and specificity are also known as True Positive Rate (TPR) and True Negatives Rate (TNR) respectively. High values for both TNR and TPR identify a good “medical test”.[11] In our context, we want to assess to what degree a positive result from our current test (the CVSS score) matches the illness (the vulnerability being actually exploited in the wild and tracked in SYM). The sensitivity and specificity measures are computed as:

$$Sensitivity = P(v\text{'s Risk factor above } t|\ v \in \mathsf{S}YM) = a/(a+c) \tag{2}$$

$$Specificity = P(v\text{'s Risk factor below } t|\ v \not\in \mathsf{S}YM) = d/(b+d) \tag{3}$$

where t is the threshold. Sensitivity and specificity outline the performance of the test in identifying exploits, but say little about its effectiveness in terms of diminished risk.

Risk Reduction. To understand the effectiveness of a policy we adopt an approach similar to that used by Evans in [Evans 1986] to estimate the effectiveness of seat belts in preventing fatalities. In his case, the “effectiveness” was given by the difference in the probability of having a fatal car crash when wearing a seatbelt and when not wearing it ($Pr(Death\ \&\ Seat\ belt\ on) - Pr(Death\ \&\ not\ Seat\ belt\ on)$).

In our case, we measure the ability of a risk factor to predict the actual exploit in the wild. Formally, the risk reduction is calculated as

$$RR = P(v \in \mathsf{S}YM|v\text{'s Risk factor above } t) - P(v \in \mathsf{S}YM|v\text{'s Risk factor below } t) \tag{4}$$

therefore $RR = a/(a+b) - c/(c+d)$. An high risk reduction identifies risk factors that clearly discern between high-risk and low-risk vulnerabilities, and are therefore good *decision variables* to act upon: the most effective strategy is identified by the risk factor with the highest risk reduction.

4.3. Data Analysis

Sensitivity and specificity. Figure 5 reports the sensitivity and specificity levels respective to different CVSS thresholds. Sensitivity is represented by the blue solid line; specificity is represented by the grey dotted line. The vertical red line outlines the CVSS threshold fixed by PCI DSS ($cvss = 4$). The green vertical line marks the threshold that separates LOW CVSS vulnerabilities from MEDIUM+HIGH CVSS vulnerabilities ($cvss = 6$).

[11] Some may prefer the False Positive Rate (FPR) to the TNR. Note that TNR=1-FPR (as in our case $d/(b+d) = 1 - b/(b+d)$). We choose to report the TNR here because 1) it has the same direction of the TPR (higher is better); 2) it facilitates the identification of the threshold with the best trade-off by intersecting TPR.

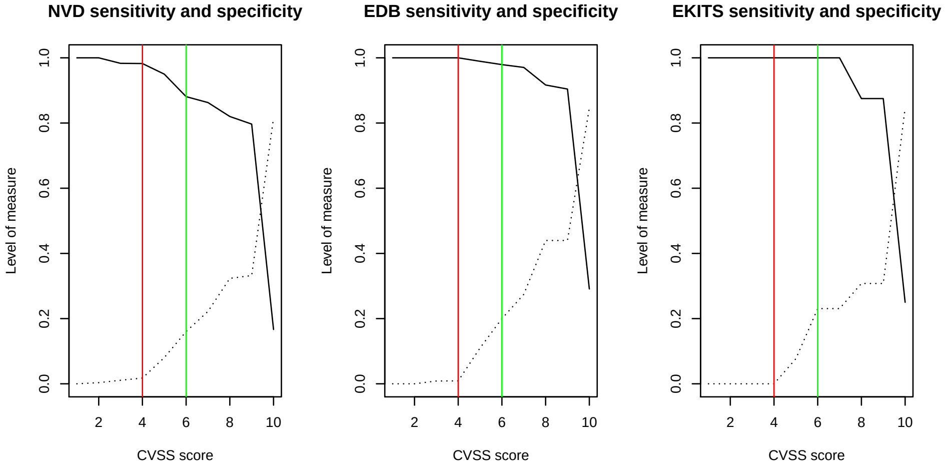


Fig. 5. Sensitivity (solid line) and specificity (dotted line) levels for different CVSS thresholds. The red line identifies the threshold for PCI DSS compliance ($cvss = 4$). The green line identifies the threshold between LOW and MEDIUM+HIGH vulnerabilities ($cvss = 6$, see histogram in Figure 1).

Unsurprisingly, low CVSS scores show a very low specificity, as most non-exploited vulnerabilities result *above* the threshold. With increasing CVSS thresholds, the specificity measure gets better without sensibly affecting sensitivity. The best trade-off obtainable with the sole CVSS score is achieved with a threshold of 8, where specificity grows over 30% and sensitivity sets at around 80%. To further increase the threshold causes the sensitivity measure to collapse. In EKITS, because most vulnerabilities in the black markets are exploited and their CVSS scores are high, the specificity measure can not significantly grow without collapsing sensitivity. In the Appendix (Table X) we report the full set of results.

Risk reduction. In Figure 6 we report our results for risk reduction (RR). The mere CVSS score (green squares), irrespectively of its threshold level, always defines a poor patching policy with very low risk reduction. The existence of a public proof-of-concept exploit is a good risk factor, yielding higher risk reduction levels (40%). The presence of an exploit in the black markets is the most effective risk factor to consider.

Table IX reports the numerical Risk Reduction for a sample of thresholds. The full list of results is available in the Appendix, Table X. A CVSS score of 6 entails a Risk Reduction of 4%; the performance is slightly better, but still unsatisfactory, if the threshold is raised to 9. Overall, CVSS' Risk Reduction stays below 10% for most thresholds. Even by considering the 95% confidence interval, we can conclude that CVSS-only based policies may be unsatisfactory from a risk-reduction point of view. Unsurprisingly, the test with the CVSS score alone results in very high p-values, that in this case testify that CVSS as a risk factor does not mark high risk vulnerabilities any better than random selection would do.

The existence of a proof-of-concept exploit improves greatly the performance of the policy: with "CVSS ≥ 6 + PoC" a RR of 42% can be achieved with very high statistical significance. This result is comparable to wearing a seat belt while driving, as those entail a reduction in risk of "only" 43% [Evans 1986]. The highest risk reduction (80%) is obtained by considering the existence of an exploit in the black markets.

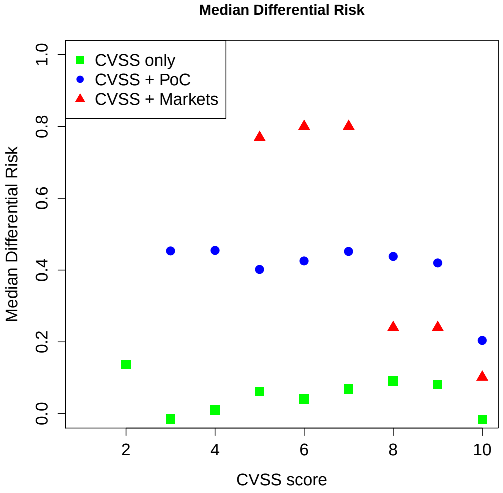


Fig. 6. Risk reduction (RR) entailed by different risk factors.

Table IX. Risk Reduction for a sample of thresholds.

Risk factor threshold	RR	95% RR conf. int.	Significance
CVSS ≥6	4%	-5% ; 12%	
CVSS ≥6 + PoC	42%	38% ; 48%	****
CVSS ≥6 + Bmar	80%	80% ; 81%	*
CVSS ≥9	8%	1% - 15%	
CVSS ≥9 + PoC	42%	36% - 49%	****
CVSS ≥9 + Bmar	24%	23% - 29%	

Note: Risk Reduction of vulnerability exploitation depending on policy and information at hand (CVSS, PoC, Markets). Significance is reported by a Bonferroni-corrected Fisher Exact test (data is sparse) for three comparison (CVSS vs CVSS+PoC vs CVSS+BMar) per experiment [Bland and Altman 1995]. A **** indicates the Bonferroni-corrected equivalent of $p < 1E-4$; *** $p < 0.001$; ** $p < 0.01$; * $p < 0.05$; nothing is reported for other values. Non-significant results indicate risk factors that perform indistinguishably at marking "high risk" vulnerabilities than random selection. The full set of results is available in Appendix, Table X.

5. DISCUSSION

We now summarize the main observations of our study. We focus on: (1) CVSS characteristics; (2) Risk reduction. Our results illustrate a mixed picture, with the margin of statistical significance varying across the main hypotheses.

(1) *The CVSS Impact submetric assumes only a few of the possible values, as Confidentiality and Integrity losses usually go hand-in-hand.* The Availability CVSS assessment adds very little variability to the score, so of the 3 dimensions of the Impact subscore, only 2 are effectively relevant.

(2) *The CVSS Exploitability metric reveals little to none variability.* The only variability among the greatest majority of vulnerabilities in NVD is given for this metric by the Access Complexity variable. Authentication and Access Vector show very little (Access Vector) to almost none (Authentication) variability. The effect of this is that the Exploitability submetric results flattened around very high values. As

a consequence, the Exploitability submetric is unsuitable as a characterization of "likelihood of exploit".

(3) *The CVSS base score alone is a poor risk factor from a statistical perspective.* Our results indicate that policies based on CVSS scores, such as the US Government NIST SCAP protocol or the world-wide used PCI DSS may not be effective in providing significant risk reductions. Our results demonstrate that using the CVSS score as a selection criterion is statistically indistinguishable from randomly picking vulnerabilities to fix.

By considering risk factors other than the sole CVSS score it may be possible to obtain more effective (and significant) strategies:

(1) The existence of a proof-of-concept exploit is an interesting risk factor to consider. PoC-based policies can entail risk reductions up to 45% of the original risk.
(2) The black markets are an even more important source of risk. Our results show that the inclusion of this risk factor can increase risk reduction up to 80%.

Our methodology is useful for both academic and industry practitioners. A case control study can be the methodology of choice when randomized trials and controlled experiments can not be performed. For example, one can not ask users to stay vulnerable and see if they get a virus or a network attack[12]. On the negative side, it has less power to determine causality than controlled experiments have, because it looks backwards rather than directly controlling an experimental process. Yet, the methodology is appropriate to evaluate the strength of the correlation between an observation of interest and some hypothesized risk factor/explanatory variable one may consider. Many of the risk factors we consider (such as CVSS, proof-of-concept exploits, etc.) are the *de-facto* standards in industry, generating a multi-million business (a casual walk among the stands of BlackHat or RSA vendors would make it immediate). Evidence of the effectiveness of these 'metrics' is however unclear, and case-control studies can be a sound scientific method to evaluate the relevance of any risk factor by using the very data that industry has available. Security data has multiple limitations that should be carefully considered when performing related studies. An overview of these problematics is given in [Christey and Martin 2013]. The most important advantage of the methodology is that it allows the researcher to *control* the different factors that may influence the outcome of the observation of interest. By design, any residual noise is evened-out by randomization in both the selection of the sample vulnerability and the bootstrapping procedure. Our results can be tailored around specific case studies by plugging in the methodology any risk factor, cost, time-to-deploy, or organizational effort that are relevant to the case.

6. THREATS TO VALIDITY

We identify a number of threats to validity [Perry et al. 2000] to our study.

Construct validity. Data collection is the main issue in an empirical study. SYM and EKITS may be particularly critical to the soundness of our conclusions. Because of the unstructured dataset of the original SYM dataset, to build SYM we needed to take some preliminary steps. A first issue is that the collected CVEs may not be relevant to the actual threat. To address this issue, we proceeded in two steps. First, we manually analyzed a random selection of about 50 entries to check for the relevance of the CVE entries to the actual attack described in the signature. An informal communication with Symantec confirmed that the CVEs are indeed relevant to the attack.

[12] Using honeynets for experiments would not give a controlled experiment either as they are artificial and not actually used.

Due to the shady nature of the tools, the list of exploited CVEs in EKITS may be not representative of the population of CVEs in the black markets; moreover, criminals may "falsely report" what CVEs their tools attack, e.g. to increase sells. To mitigate the problem, we crossed-referenced EKITS entries with knowledge from the security research community and from our direct testing of tools traded in the the black markets [Allodi et al. 2013].

External validity. is concerned with the applicability of our results to real-world scenarios. Symantec is a world-wide company and a leader in the security industry. We are therefore confident in considering their data as representative of real-world attack scenarios. Yet, our conclusion can not be generalized to targeted attacks. These attacks in the wild usually target a specific platform or system and are less likely to generate an entry in a general purpose anti-virus product.

An important point to address is that our approach does not address *changing behavior* of the attacker. For example, if all vulnerabilities from the black markets with a certain characteristic get patched, the attacker may simply modify his own attack strategy in such a way to render the defender's strategy ineffective. This is a common problem in any game-theoretical approach: unfortunately the defender ought to move first, thus the attacker can always adapt to the defender's strategy (hence the definition of *equilibrium* as the state of the game into which neither attacker nor defender have a good reason to change their strategy). This problem is present in the application of any security technology or solution available. The game-theoretic nature of the problem is not addressed by our methodology either. We reserve the exploration of this issue for further work.

7. RELATED WORK

Vulnerability studies. Several studies before ours have dealt with software vulnerabilities, software risk and risk mitigation. Among all, Frei et al. [Frei et al. 2006] were maybe the first to link the idea of life-cycle of a vulnerability to the patching process. Their dataset was a composition of NVD, the Open Source Vulnerabiltity DataBase and 'FVDB' (Frei's Vulnerability DataBase, obtained from the examination of security advisories for patches). The life-cycle of a vulnerability includes discovery time, exploitation time and patching time. They showed that exploits are often quicker to arrive than patches are. They were the first to look, in particular, at the difference in time between time of first "exploit" and time of disclosure of the vulnerability. This work have recently been extended by Shahzad et al. [Shahzad et al. 2012], who presented a comprehensive vulnerability study on NVD and OSVDB datasets (and Frei's) that included vendors and software in the analysis. Many descriptive trends in timings of vulnerability patching and exploitation are presented. However, their use of exploit data from OSVDB says little about the actual exploitation of a vulnerability [Christey and Martin 2013]. NVD timing data has also been reported to generate an unforeseeable amount of noise because of how the vulnerability disclosure process works [Schryen 2009; Christey and Martin 2013]. To avoid these problems we make an effort in finding data on actual exploits, proof-of-concept exploits, and exploits in the black markets. We provide a descriptive analysis of this vulnerability data, and use our findings to provide advices to practitioners that desire to assess software vulnerability risk and efficacy of remediation strategies. For a thorough description of our datasets and a preliminary discussion on the data, see [Allodi and Massacci 2012]; for additional details on Symantec's attack data we point the reader to [Dumitras and Shou 2011].

The idea of using vulnerability data to assess overall security is not new by itself. Attack surfaces [Manadhata and Wing 2011] and attack graphs [Wang et al. 2008] are seminal approaches to the problem: the former uses vulnerability data to compute an "exposure metric" of the vulnerable systems to potential attacks; the latter aims at modeling consequent attacks on a system (or network of systems) the attacker might perpetrate to reach a (usually critical) component such as a data server. These approaches however lack of a characterization of vulnerability risk. Our methodology integrates these approaches by providing a risk estimation for vulnerabilities; our results can be plugged in both attack graphs and attack surface estimations to obtain more precise assessments.

CVSS. An analysis of the distribution of CVSS scores and subscores has been presented by Scarfone et al. [Scarfone and Mell 2009] and Gallon [Gallon 2011]. However, while including CVSS subscore analysis, their results are limited to data from NVD and do not provide any insight on vulnerability exploitation. In this sense, Bozorgi et al. [Bozorgi et al. 2010] were probably the first to look for this correlation. They showed that the CVSS characterization of "likelihood to exploit" did not match with data on proof-of-concept exploits in EDB. We extended their first observation with a in-depth analysis of subscores and of actual exploitation data.

Vulnerability models. Other studies focused on the modeling of the vulnerability discovery processes, which arguably lays the ground for the "vulnerability remediation" process, focus of our work. As noted by [Shin and Williams 2013], vulnerability models can help "*security engineers to prioritize security inspection and testing efforts*" by, for example, identifying software components that are most susceptible to attacks [Gegick et al. 2009] or most likely to have unknown vulnerabilities hidden in the code [Neuhaus et al. 2007]. Our contribution differs, in general, from work on vulnerability models in that we do not aim at identifying "vulnerable components" or previously unknown vulnerabilities to point software engineers in the right direction. We instead propose a methodology to evaluate the risk of already known vulnerabilities to be exploited in the wild, and therefore may need immediate remediation or mitigation on the deployment side rather than on the development side.

Alhazmi et al.'s [Alhazmi and Malaiya 2008] and Ozment's [Ozment 2007] work are both central in vulnerability discovery models research. Alhazmi et al. fit six vulnerability models to vulnerability data of four major operative systems, and show that Alhazmi's 'S shaped' model is the one that performs the best. [Shin and Williams 2013] suggest that vulnerability models might be substituted with fault prediction models, and showed that performances in terms of "recall" and "precision" do not differ sensibly between the two. However, as previously underlined by Ozment [Ozment 2007], vulnerability models may rely on unsound assumptions such as the independence of vulnerability discoveries. Current vulnerability discovery models are indeed not general enough to represent trends for all software [Massacci and Nguyen 2012]. Moreover, vulnerability disclosure and discovery are complex processes [Ozment 2005; Clark et al. 2010], and can be influenced by {black/white}-hat community activities [Clark et al. 2010] and economics [Miller 2007].

Markets for vulnerabilities. Our analysis of vulnerabilities traded in the black markets is also interesting because it supports the hypothesis that the exploit markets are significantly different (and more stable) than the previous IRC markets frequented by cyber criminals were [Herley and Florencio 2010]. Previous work from the authors of this manuscript also experimentally showed that the goods traded in the black markets are very reliable in delivering attacks and are resilient to aging [Allodi et al. 2013].

8. CONCLUSION

In this paper we have proposed the case-control study methodology as an operative framework for security studies. In a case-control study the researcher looks backward at some of the *cases* (for example vulnerabilities exploited in the wild) and compare them with *controls* (in our case randomly selected vulnerabilities with similar characteristics such as year of discovery or software type). The purpose is to identify whether some *risk factor* (in our scenario a high CVSS score, or the existence of a proof of concept exploit) is a good explanation of the cases and therefore represents a decision variable upon which the system administrator must act.

To illustrate the methodology we first analyzed the CVSS score in its capability to express the Impact and the Likelihood of an exploitation to happen. We showed that a proper characterization of 'likelihood of exploit' is not present in the CVSS score. We then evaluated its performances as a "risk indicator" [Council 2010; Quinn et al. 2010] by performing a case-control study, in which we sample the data at hand to test how the CVSS score correlates with exploitation in the wild. Our results show that the CVSS base score never achieves high rates of identified true positives (sensitivity) simultaneously with a high rate of true negatives (specificity).

Finally, we showed how the methodology can be used to evaluate the 'effectiveness' of multiple policies that consider different risk factors. Our results show that the sole CVSS score performs no better than 'randomly picking' vulnerabilities to fix and may lead to negligible risk reductions. Markedly better results can instead be obtained when additional risk factors are considered; in this study we considered the existence of a proof-of-concept exploit and of an exploit traded in the black markets.

In future work we plan to integrate our methodology with additional evaluation factors such as the cost of a strategy or the criticality of the assets. Another interesting venue would be to apply our methodology to other domains (e.g. critical infrastructures and targeted attacks).

REFERENCES

O.H. Alhazmi and Y.K. Malaiya. 2008. Application of Vulnerability Discovery Models to Major Operating Systems. *IEEE Transactions on Reliability* 57, 1 (march 2008), 14 –22. DOI:http://dx.doi.org/10.1109/TR.2008.916872

Luca Allodi, Vadim Kotov, and Fabio Massacci. 2013. MalwareLab: Experimentation with Cybercrime attack tools. In *Proceedings of the 2013 6th Workshop on Cybersecurity Security and Test*.

Luca Allodi and Fabio Massacci. 2012. A Preliminary Analysis of Vulnerability Scores for Attacks in Wild. In *Proceedings of the 2012 ACM CCS Workshop on Building Analysis Datasets and Gathering Experience Returns for Security*.

J Martin Bland and Douglas G Altman. 1995. Multiple significance tests: the Bonferroni method. 310, 6973 (1995), 170.

Mehran Bozorgi, Lawrence K. Saul, Stefan Savage, and Geoffrey M. Voelker. 2010. Beyond heuristics: learning to classify vulnerabilities and predict exploits. In *Proceedings of the 16th ACM International Conference on Knowledge Discovery and Data Mining*. ACM, 105–114.

Steve Christey and Brian Martin. 2013. Buying into the bias: why vulnerability statistics suck. https://www.blackhat.com/us-13/archives.html#Martin. (July 2013).

Sandy Clark, Stefan Frei, Matt Blaze, and Jonathan Smith. 2010. Familiarity breeds contempt: the honeymoon effect and the role of legacy code in zero-day vulnerabilities. In *Proceedings of the 26th Annual Computer Security Applications Conference*. 251–260. http://doi.acm.org/10.1145/1920261.1920299

PCI Council. 2010. PCI DSS Requirements and Security Assessment Procedures, Version 2.0. (2010). https://www.pcisecuritystandards.org/documents/pci_dss_v2.pdf

Richard Doll and A Bradford Hill. 1950. Smoking and Carcinoma of the Lung. *British Medical Journal* 2, 4682 (1950), 739–748.

Tudor Dumitras and Petros Efstathopoulos. 2012. Ask WINE: are we safer today? evaluating operating system security through big data analysis. In *Proceeding of the 2012 USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET'12)*. 11–11.

Tudor Dumitras and Darren Shou. 2011. Toward a standard benchmark for computer security research: The Worldwide Intelligence Network Environment (WINE). In *Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security*. ACM, 89–96.

L. Evans. 1986. The effectiveness of safety belts in preventing fatalities. *Accident Analysis & Prevention* 18, 3 (1986), 229–241.

Stefan Frei, Martin May, Ulrich Fiedler, and Bernhard Plattner. 2006. Large-scale vulnerability analysis. In *Proceedings of the 2006 SIGCOMM workshop on Large-scale attack defense*. ACM, 131–138.

L. Gallon. 2011. Vulnerability Discrimination Using CVSS Framework. In *Proceedings of the 4th IFIP International Conference on New Technologies, Mobility and Security*. 1–6.

Michael Gegick, Pete Rotella, and Laurie A. Williams. 2009. Predicting Attack-prone Components. In *Proceedings of the 2nd International Conference on Software Testing Verification and Validation (ICST'09)*. 181–190.

Chris Grier, Lucas Ballard, Juan Caballero, Neha Chachra, Christian J. Dietrich, Kirill Levchenko, Panayiotis Mavrommatis, Damon McCoy, Antonio Nappa, Andreas Pitsillidis, Niels Provos, M. Zubair Rafique, Moheeb Abu Rajab, Christian Rossow, Kurt Thomas, Vern Paxson, Stefan Savage, and Geoffrey M. Voelker. 2012. Manufacturing compromise: the emergence of exploit-as-a-service. In *Proceedings of the 19th ACM Conference on Computer and Communications Security*. ACM, 821–832.

C. Herley and D. Florencio. 2010. Nobody sells gold for the price of silver: Dishonesty, uncertainty and the underground economy. *Economics of Information Security and Privacy* (2010).

Siv Hilde Houmb, Virginia NL Franqueira, and Erlend A Engum. 2010. Quantifying security risk level from CVSS estimates of frequency and impact. 83, 9 (2010), 1622–1634.

Vadim Kotov and Fabio Massacci. 2013. Anatomy of Exploit Kits. Preliminary Analysis of Exploit Kits as Software Artefacts. In *Proc. of ESSoS 2013*.

Pratyusa K. Manadhata and Jeannette M. Wing. 2011. An Attack Surface Metric. *IEEE Transactions on Software Engineering* 37 (2011), 371–386. DOI:http://dx.doi.org/10.1109/TSE.2010.60

Fabio Massacci, Stephan Neuhaus, and Viet Nguyen. 2011. After-Life Vulnerabilities: A Study on Firefox Evolution, Its Vulnerabilities, and Fixes. In *Proceedings of the 2011 Engineering Secure Software and Systems Conference (ESSoS'11) (Lecture Notes in Computer Science)*. 195–208.

Fabio Massacci and Viet Nguyen. 2012. An Independent Validation of Vulnerability Discovery Models. In *Proceeding of the 7th ACM Symposium on Information, Computer and Communications Security (ASIACCS'12)*.

Peter Mell, Karen Scarfone, and Sasha Romanosky. 2007. *A Complete Guide to the Common Vulnerability Scoring System Version 2.0*. Technical Report. FIRST, Available at http://www.first.org/cvss.

C. Miller. 2007. The legitimate vulnerability market: Inside the secretive world of 0-day exploit sales. In *Proceedings of the 6th Workshop on Economics and Information Security*.

Stephan Neuhaus, Thomas Zimmermann, Christian Holler, and Andreas Zeller. 2007. Predicting Vulnerable Software Components. In *Proceedings of the 14th ACM Conference on Computer and Communications Security*. 529–540.

A. Ozment. 2005. The likelihood of vulnerability rediscovery and the social utility of vulnerability hunting. In *Proceedings of the 4th Workshop on Economics and Information Security*.

Andy Ozment. 2007. Improving vulnerability discovery models. In *Proceedings of the 3rd Workshop on Quality of Protection*. 6–11.

Dewayne E. Perry, Adam A. Porter, and Lawrence G. Votta. 2000. Empirical studies of software engineering: a roadmap. In *Proceedings of the 22nd Conference on The Future of Software Engineering*. ACM, 345–355.

Stephen D. Quinn, Karen A. Scarfone, Matthew Barrett, and Christopher S. Johnson. 2010. *SP 800-117. Guide to Adopting and Using the Security Content Automation Protocol (SCAP) Version 1.0*. Technical Report.

R Core Team. 2012. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. http://www.R-project.org ISBN 3-900051-07-0.

Karen Scarfone and Peter Mell. 2009. An analysis of CVSS version 2 vulnerability scoring. In *Proceedings of the 3rd International Symposium on Empirical Software Engineering and Measurement*. 516–525.

Guido Schryen. 2009. A Comprehensive and Comparative Analysis of the Patching Behavior of Open Source and Closed Source Software Vendors. In *Proceedings of the 2009 Fifth International Conference on IT Security Incident Management and IT Forensics (IMF '09)*. IEEE Computer Society, Washington, DC, USA, 153–168. DOI:http://dx.doi.org/10.1109/IMF.2009.15

Muhammad Shahzad, Muhammad Zubair Shafiq, and Alex X. Liu. 2012. A large scale exploratory analysis of software vulnerability life cycles. In *Proceedings of the 34th International Conference on Software Engineering*. IEEE Press, 771–781.

Yonghee Shin and Laurie Williams. 2013. Can traditional fault prediction models be used for vulnerability prediction? *Empirical Software Engineering* 18, 1 (2013), 25–59. DOI:http://dx.doi.org/10.1007/s10664-011-9190-8

Lingyu Wang, Tania Islam, Tao Long, Anoop Singhal, and Sushil Jajodia. 2008. An Attack Graph-Based Probabilistic Security Metric. In *Proceedings of the 22nd IFIP WG 11.3 Working Conference on Data and Applications Security*. Lecture Notes in Computer Science, Vol. 5094. Springer Berlin / Heidelberg, 283–296.

Branden R Williams and Anton Chuvakin. 2012. *PCI Compliance: Understand and implement effective PCI data security standard compliance*. Syngress Elsevier.

Received ; revised ; accepted

Online Appendix to: Comparing vulnerability severity and exploits using case-control studies.

LUCA ALLODI, University of Trento
FABIO MASSACCI, University of Trento

A. DETAILED ANALYSIS TABLE

In this Appendix we report the full set of results from the bootstrapped random sampling described in Section 4. Table X reports the results for all the active controls.

Sensitivity, specificity and risk reduction are calculated on the median of the results obtained from the bootstrapping procedure. The last column of the table reports the median p-value of the results: the reported p-value is the value under which 50% of our samples are. We report the results numerically as they are in output from our experiment. The level of significance should however be corrected (e.g. by means of a Bonferroni correction) for each experimental run, when comparing the results for a particular combination of controls [Bland and Altman 1995].

Additional results for different combinations of controls are reported in Table XI and Table XII. The full set of results is available upon request to the authors.

The control variables we vary across the experiments are:

- censorYr: only CVEs disclosed in the years 2009,2010,2011,2012 are considered in the study.
- censorSw: only CVEs affecting software also included in SYM are considered.
- checkYr: for each CVE in SYM another is picked for inclusion in the sample that has the *exact same year* as the one in SYM. This is different from censorYr as in that case the only constraint is that the chosen vulnerability lays in the 2009-2012 time range.
- checkSw (available upon request): each CVE picked for the sample has exactly one counterpart, software-wise, in SYM.

Our risk factors are:

- CVSS score: the vulnerability has been disclosed (i.e. we consider every vulnerability) and has a high CVSS threshold.
- Proof-of-concept exploit: the vulnerability has a proof-of-concept exploit in EDBand has a high CVSS threshold.
- Black-marketed exploit: an exploit is reported to be traded in the black markets for that vulnerability and has a high CVSS threshold.

DOI:http://dx.doi.org/10.1145/0000000.0000000

Table X. Full result table for the experiment reported in this manuscript. We report the p-values as are. When applying corrections like the Bonferroni, only the rows that belong to the same experiment run should be considered, i.e. rows reporting the same combination of controls and threshold.

db	censorYr	censorSw	checkYr	checkCIA	threshold	Above & $\in$ SYM	Below & $\in$ SYM	Above & $\not\in$ SYM	Below & $\not\in$ SYM	Sensitivity	Specificity	Risk Reduction	95 conf. Int.	p-value
nvd	✓	✓	✓	✓	1	53	0	281	0	100.00%	0.00%	NA	NA	1.00E+00
nvd	✓	✓	✓	✓	2	54	0	279	1	100.00%	0.35%	13.69%	-85.7%;19.37%	1.00E+00
nvd	✓	✓	✓	✓	3	52	1	277	3	98.31%	1.09%	-1.47%	-47.07%;18.81%	6.22E-01
nvd	✓	✓	✓	✓	4	53	1	274.5	5	98.25%	1.76%	1.09%	-35.03%;19.14%	6.06E-01
nvd	✓	✓	✓	✓	5	51	3	258	23	95.00%	8.04%	6.12%	-7.12%;16.09%	5.52E-01
nvd	✓	✓	✓	✓	6	47	6	235	45	88.10%	16.03%	4.14%	-4.29%;12.38%	5.08E-01
nvd	✓	✓	✓	✓	7	47	7	218	62	86.27%	22.22%	6.91%	-0.71%;14.57%	1.96E-01
nvd	✓	✓	✓	✓	8	44	10	190	91	82.00%	32.35%	9.16%	2.04%;15.02%	4.11E-02
nvd	✓	✓	✓	✓	9	42	11	187	93	79.67%	33.22%	8.09%	0.98%;15.14%	7.61E-02
nvd	✓	✓	✓	✓	10	9	44	52	228	16.67%	81.54%	-1.60%	-10.73%;7.1%	5.68E-01
edb	✓	✓	✓	✓	1	95	0	115	0	100.00%	0.00%	NA	NA	1.00E+00
edb	✓	✓	✓	✓	2	95	0	115	0	100.00%	0.00%	NA	NA	1.00E+00
edb	✓	✓	✓	✓	3	95	0	114.5	1	100.00%	0.86%	45.39%	41.46%;49.23%	1.00E+00
edb	✓	✓	✓	✓	4	95	0	114	1	100.00%	0.89%	45.41%	42.18%;49.76%	1.00E+00
edb	✓	✓	✓	✓	5	93	1	103	13	98.94%	11.01%	40.21%	35.13%;45.36%	3.81E-03
edb	✓	✓	✓	✓	6	93	2	92	23	97.92%	20.00%	42.58%	37.86%;48.25%	2.90E-05
edb	✓	✓	✓	✓	7	92	3	84	32	97.06%	27.43%	45.18%	39.55%;51.16%	3.56E-07
edb	✓	✓	✓	✓	8	87	8	64	51	91.67%	43.97%	43.77%	37.56%;50.64%	3.57E-09
edb	✓	✓	✓	✓	9	86	9	65	51	90.43%	43.97%	42.00%	36.25%;49.11%	2.07E-08
edb	✓	✓	✓	✓	10	28	68	18	98	29.10%	84.62%	20.46%	12.53%;29.9%	1.81E-02
ekits	✓	✓	✓	✓	1	40	0	13	0	100.00%	0.00%	NA	NA	1.00E+00
ekits	✓	✓	✓	✓	2	40	0	13	0	100.00%	0.00%	NA	NA	1.00E+00
ekits	✓	✓	✓	✓	3	40	0	13	0	100.00%	0.00%	NA	NA	1.00E+00
ekits	✓	✓	✓	✓	4	40	0	13	0	100.00%	0.00%	NA	NA	1.00E+00
ekits	✓	✓	✓	✓	5	40	0	12	1	100.00%	7.69%	76.92%	76.47%;78.43%	2.45E-01
ekits	✓	✓	✓	✓	6	40	0	10	3	100.00%	23.08%	80.00%	79.59%;81.25%	1.22E-02
ekits	✓	✓	✓	✓	7	40	0	10	3	100.00%	23.08%	80.00%	79.59%;80%	1.22E-02
ekits	✓	✓	✓	✓	8	35	5	9	4	87.50%	30.77%	23.99%	23.51%;29.55%	1.99E-01
ekits	✓	✓	✓	✓	9	35	5	9	4	87.50%	30.77%	23.99%	23.51%;29.55%	1.99E-01
ekits	✓	✓	✓	✓	10	10	30	2	11	25.00%	84.62%	10.16%	8.65%;10.83%	7.07E-01

Table XI. Table of results for the bootstrapped case-control study for the CIA, censor year, censor software controls.

db	censorYr	censorSw	checkYr	checkCIA	threshold	Above & ∈ SYM	Below & ∈ SYM	Above & ∉ SYM	Below & ∉ SYM	Sensitivity	Specificity	Risk Reduction	95 conf. Int.	p-value
nvd				✓	1	54	0	1118	0	100.00%	0.00%	NA	NA	1.00E+00
nvd				✓	2	54	0	1115	3	100.00%	0.27%	4.63%	-21.06%;5.75%	1.00E+00
nvd				✓	3	55	0	1094	23	100.00%	2.05%	4.62%	-3.01%;5.9%	6.28E-01
nvd				✓	4	54	0	1087	30	100.00%	2.69%	4.39%	-1.18%;5.86%	6.24E-01
nvd				✓	5	55	1	1009	107	98.56%	9.58%	4.63%	2.53%;6.19%	2.48E-02
nvd				✓	6	51	4	857	260	92.68%	23.29%	4.11%	2.01%;6.05%	4.19E-03
nvd				✓	7	50	5	739	378	90.24%	33.84%	4.86%	2.66%;6.72%	8.89E-05
nvd				✓	8	43	12	424	694	77.78%	62.08%	7.52%	4.98%;9.82%	4.15E-09
nvd				✓	9	42	13	416	701	77.08%	62.76%	7.37%	4.95%;10.16%	7.77E-09
nvd				✓	10	13	42	220	896	23.91%	80.27%	1.14%	-1.68%;4.32%	4.10E-01
edb				✓	1	105	0	937	0	100.00%	0.00%	NA	NA	1.00E+00
edb				✓	2	105	0	937	0	100.00%	0.00%	10.07%	8.71%;11.43%	1.00E+00
edb				✓	3	104	0	930	9	100.00%	0.96%	9.58%	-6.62%;11.39%	1.00E+00
edb				✓	4	105	0	925	13	100.00%	1.38%	9.66%	-3.3%;11.61%	6.19E-01
edb				✓	5	104	1	866	72	99.07%	7.71%	9.62%	6.38%;11.68%	4.53E-03
edb				✓	6	102	3	720	216.5	97.18%	23.09%	10.98%	8.62%;13.11%	2.83E-08
edb				✓	7	100	4	630	307	95.92%	32.69%	12.27%	10%;14.47%	6.12E-12
edb				✓	8	91	13	340	597	87.50%	63.72%	18.99%	15.95%;21.83%	1.74E-24
edb				✓	9	90	14	336	603	86.54%	64.19%	18.96%	15.71%;21.75%	2.88E-24
edb				✓	10	35	69	135	801	33.68%	85.57%	12.98%	7.47%;17.97%	3.24E-06
ekits				✓	1	77	0	25	0	100.00%	0.00%	NA	NA	1.00E+00
ekits				✓	2	77	0	25	0	100.00%	0.00%	NA	NA	1.00E+00
ekits				✓	3	77	0	25	0	100.00%	0.00%	NA	NA	1.00E+00
ekits				✓	4	77	0	25	0	100.00%	0.00%	NA	NA	1.00E+00
ekits				✓	5	77	0	22	3	100.00%	12.00%	77.78%	77.78%;77.78%	1.34E-02
ekits				✓	6	75	2	17	8	97.40%	32.00%	61.52%	61.52%;61.52%	1.56E-04
ekits				✓	7	71	6	16	9	92.21%	36.00%	41.61%	41.61%;41.61%	1.59E-03
ekits				✓	8	60	17	14	11	77.92%	44.00%	20.37%	20.37%;20.37%	4.13E-02
ekits				✓	9	60	17	14	11	77.92%	44.00%	20.37%	20.37%;20.37%	4.13E-02
ekits				✓	10	14	63	3	22	18.18%	88.00%	8.24%	8.24%;8.24%	5.54E-01
nvd	✓			✓	1	48.5	0	1077.5	0	100.00%	0.00%	NA	NA	1.00E+00
nvd	✓			✓	2	47	0	1075	4	100.00%	0.37%	4.04%	-29.13%;5.13%	1.00E+00
nvd	✓			✓	3	48	0	1062	15	100.00%	1.40%	4.01%	-7.62%;5.33%	1.00E+00
nvd	✓			✓	4	48	0	1051	26	100.00%	2.45%	4.12%	-2.32%;5.41%	6.31E-01
nvd	✓			✓	5	47	1	961	117	97.87%	10.81%	3.75%	1.35%;5.62%	6.01E-02
nvd	✓			✓	6	46	3	834	245	94.34%	22.77%	4.11%	2.22%;5.77%	2.65E-03
nvd	✓			✓	7	45.5	3	695	383	93.33%	35.48%	5.31%	3.19%;6.84%	6.88E-06
nvd	✓			✓	8	43	6	443	634	88.46%	58.92%	7.84%	5.79%;10.03%	3.61E-11
nvd	✓			✓	9	42	6	435	642	87.61%	59.58%	7.95%	5.78%;10.34%	3.50E-11
nvd	✓			✓	10	10	38	167	911	20.75%	84.52%	1.64%	-1.67%;4.84%	3.27E-01
edb	✓			✓	1	86	0	795	0	100.00%	0.00%	NA	NA	1.00E+00
edb	✓			✓	2	88	0	794	0	100.00%	0.00%	NA	NA	1.00E+00
edb	✓			✓	3	87	0	789	4	100.00%	0.51%	9.92%	8.79%;11.02%	1.00E+00
edb	✓			✓	4	87	0	786	8	100.00%	1.01%	10.01%	8.77%;11.04%	1.00E+00
edb	✓			✓	5	87	0	714	80	100.00%	10.10%	10.61%	8.9%;11.95%	4.21E-04
edb	✓			✓	6	86	0	604	190	100.00%	23.90%	12.23%	10.49%;13.91%	1.74E-09
edb	✓			✓	7	87	1	525	269	98.91%	33.76%	13.87%	12.19%;15.38%	3.94E-14
edb	✓			✓	8	83	4	268	526	95.45%	66.28%	22.82%	19.79%;25.57%	3.42E-31
edb	✓			✓	9	82	4	264	529	95.29%	66.75%	23.05%	20.2%;25.69%	3.55E-31
edb	✓			✓	10	26	61	66	728	29.76%	91.76%	20.49%	14.81%;26.04%	8.28E-08
ekits	✓			✓	1	40	0	13	0	100.00%	0.00%	NA	NA	1.00E+00
ekits	✓			✓	2	40	0	13	0	100.00%	0.00%	NA	NA	1.00E+00
ekits	✓			✓	3	40	0	13	0	100.00%	0.00%	NA	NA	1.00E+00
ekits	✓			✓	4	40	0	13	0	100.00%	0.00%	NA	NA	1.00E+00
ekits	✓			✓	5	40	0	12	1	100.00%	7.69%	76.92%	76.92%;76.92%	2.45E-01
ekits	✓			✓	6	40	0	10	3	100.00%	23.08%	80.00%	80%;80%	1.22E-02
ekits	✓			✓	7	40	0	10	3	100.00%	23.08%	80.00%	80%;80%	1.22E-02
ekits	✓			✓	8	35	5	9	4	87.50%	30.77%	23.99%	23.99%;23.99%	1.99E-01
ekits	✓			✓	9	35	5	9	4	87.50%	30.77%	23.99%	23.99%;23.99%	1.99E-01
ekits	✓			✓	10	10	30	2	11	25.00%	84.62%	10.16%	10.16%;10.16%	7.07E-01

Table XII. –continuation from previous page

db	censorYr	censorSw	checkYr	checkCIA	threshold	Above & $\in$ SYM	Below & $\in$ SYM	Above & $\not\in$ SYM	Below & $\not\in$ SYM	Sensitivity	Specificity	Risk Reduction	95 conf. Int.	p-value
nvd		✓		✓	1	162	0	954	0	100.00%	0.00%	NA	NA	1.00E+00
nvd		✓		✓	2	163	0	948	4	100.00%	0.41%	14.40%	-33.07%;16.46%	1.00E+00
nvd		✓		✓	3	162	1	924	29	99.39%	3.01%	11.63%	2.86%;16.37%	1.07E-01
nvd		✓		✓	4	162	1	917	36	99.36%	3.80%	11.83%	4.03%;16.22%	5.01E-02
nvd		✓		✓	5	161.5	4	828	124	97.80%	13.02%	13.47%	10.15%;16.57%	5.71E-06
nvd		✓		✓	6	144	19	691	263	88.55%	27.54%	10.71%	6.65%;14.14%	3.83E-06
nvd		✓		✓	7	137	26	605	349	84.11%	36.53%	11.55%	8.07%;15.52%	7.01E-08
nvd		✓		✓	8	99	65	386	568	60.51%	59.46%	10.13%	6.2%;13.79%	2.59E-06
nvd		✓		✓	9	98	66	381	573	59.76%	60.01%	10.10%	6.26%;13.97%	3.16E-06
nvd		✓		✓	10	32	131	156	798.5	19.40%	83.61%	2.70%	-1.56%;8.36%	3.45E-01
edb		✓		✓	1	263	0	401	0	100.00%	0.00%	NA	NA	1.00E+00
edb		✓		✓	2	262	0	401	0	100.00%	0.00%	NA	NA	1.00E+00
edb		✓		✓	3	261	1	395	6	99.62%	1.44%	24.87%	5.77%;40.96%	2.96E-01
edb		✓		✓	4	262	1	391	9	99.62%	2.23%	30.08%	16.92%;41.29%	8.93E-02
edb		✓		✓	5	258	5	353	49	98.08%	12.29%	32.91%	29.09%;37.27%	4.47E-07
edb		✓		✓	6	250	13	284	117	94.92%	29.17%	36.56%	32.49%;40.18%	3.87E-16
edb		✓		✓	7	238	24	239	161	90.71%	40.35%	36.93%	33.11%;40.28%	6.64E-20
edb		✓		✓	8	174	89	140	260	66.41%	64.99%	30.21%	26.16%;33.73%	2.14E-15
edb		✓		✓	9	173	90	139	262	65.65%	65.33%	29.80%	25.66%;33.45%	4.01E-15
edb		✓		✓	10	67	196	56	344	25.47%	86.01%	18.05%	13.12%;22.85%	2.95E-04
ekits		✓		✓	1	77	0	20	0	100.00%	0.00%	NA	NA	1.00E+00
ekits		✓		✓	2	77	0	20	0	100.00%	0.00%	NA	NA	1.00E+00
ekits		✓		✓	3	77	0	20	0	100.00%	0.00%	NA	NA	1.00E+00
ekits		✓		✓	4	77	0	20	0	100.00%	0.00%	NA	NA	1.00E+00
ekits		✓		✓	5	77	0	18	2	100.00%	10.00%	81.05%	81.05%;81.05%	4.08E-02
ekits		✓		✓	6	75	2	14	6	97.40%	30.00%	59.27%	59.27%;59.27%	8.27E-04
ekits		✓		✓	7	71	6	13	7	92.21%	35.00%	38.37%	38.37%;38.37%	4.53E-03
ekits		✓		✓	8	60	17	11	9	77.92%	45.00%	19.12%	19.12%;19.12%	5.00E-02
ekits		✓		✓	9	60	17	11	9	77.92%	45.00%	19.12%	19.12%;19.12%	5.00E-02
ekits		✓		✓	10	14	63	3	17	18.18%	85.00%	3.60%	3.6%;3.6%	1.00E+00
nvd	✓	✓		✓	1	106	0	855	0	100.00%	0.00%	NA	NA	1.00E+00
nvd	✓	✓		✓	2	105	1	849	5	99.11%	0.60%	-1.38%	-29.36%;12.47%	6.06E-01
nvd	✓	✓		✓	3	106	1	834	19	98.97%	2.17%	4.51%	-6.5%;11.9%	7.12E-01
nvd	✓	✓		✓	4	104.5	2	827	27	98.20%	3.20%	5.20%	-2.95%;11.83%	5.67E-01
nvd	✓	✓		✓	5	101	5	721	133	95.00%	15.59%	8.53%	5.14%;11.96%	1.77E-03
nvd	✓	✓		✓	6	96	11	623	232	89.91%	27.13%	8.84%	5.69%;11.62%	6.56E-05
nvd	✓	✓		✓	7	92	14	514	339	86.41%	39.62%	11.05%	8.35%;14.03%	2.44E-08
nvd	✓	✓		✓	8	79	28	391	464	73.21%	54.28%	10.99%	7.87%;14.26%	5.71E-08
nvd	✓	✓		✓	9	78	29	386	468	72.82%	54.79%	11.03%	7.7%;14.1%	6.38E-08
nvd	✓	✓		✓	10	18	88	110	745	17.00%	87.19%	3.56%	-1.47%;9.54%	2.41E-01
edb	✓	✓		✓	1	133	0	209	0	100.00%	0.00%	NA	NA	1.00E+00
edb	✓	✓		✓	2	133	0	209	0	100.00%	0.00%	NA	NA	1.00E+00
edb	✓	✓		✓	3	133	0	207	2	100.00%	0.95%	39.10%	37.49%;40.48%	5.25E-01
edb	✓	✓		✓	4	133	0	205	4	100.00%	1.92%	39.41%	38.02%;40.85%	1.61E-01
edb	✓	✓		✓	5	132	1	170	38	99.25%	18.38%	41.08%	39.25%;42.83%	3.75E-08
edb	✓	✓		✓	6	131	2	141	68	98.50%	32.39%	45.29%	43.32%;47.06%	1.69E-14
edb	✓	✓		✓	7	131	3	125	84	97.76%	40.05%	47.53%	45.98%;49.35%	5.23E-18
edb	✓	✓		✓	8	120	14	92	117	89.63%	55.95%	45.99%	43.96%;48.14%	1.05E-18
edb	✓	✓		✓	9	118	15	92	117	88.89%	55.89%	45.04%	42.87%;47.14%	5.48E-18
edb	✓	✓		✓	10	37	96	24	185	28.15%	88.68%	27.24%	23.57%;31.78%	1.17E-04
ekits	✓	✓		✓	1	40	0	13	0	100.00%	0.00%	NA	NA	1.00E+00
ekits	✓	✓		✓	2	40	0	13	0	100.00%	0.00%	NA	NA	1.00E+00
ekits	✓	✓		✓	3	40	0	13	0	100.00%	0.00%	NA	NA	1.00E+00
ekits	✓	✓		✓	4	40	0	13	0	100.00%	0.00%	NA	NA	1.00E+00
ekits	✓	✓		✓	5	40	0	12	1	100.00%	7.69%	76.92%	76.92%;76.92%	2.45E-01
ekits	✓	✓		✓	6	40	0	10	3	100.00%	23.08%	80.00%	80%;80%	1.22E-02
ekits	✓	✓		✓	7	40	0	10	3	100.00%	23.08%	80.00%	80%;80%	1.22E-02
ekits	✓	✓		✓	8	35	5	9	4	87.50%	30.77%	23.99%	23.99%;23.99%	1.99E-01
ekits	✓	✓		✓	9	35	5	9	4	87.50%	30.77%	23.99%	23.99%;23.99%	1.99E-01
ekits	✓	✓		✓	10	10	30	2	11	25.00%	84.62%	10.16%	10.16%;10.16%	7.07E-01