Can the Common Vulnerability Scoring System be Trusted? A Bayesian Analysis

Pontus Johnson, Robert Lagerström, Mathias Ekstedt, and Ulrik Franke

Abstract—The Common Vulnerability Scoring System (CVSS) is the state-of-the art system for assessing software vulnerabilities. However, it has been criticized for lack of validity and practitioner relevance. In this paper, the credibility of the CVSS scoring data found in five leading databases – NVD, X-Force, OSVDB, CERT-VN, and Cisco – is assessed. A Bayesian method is used to infer the most probable true values underlying the imperfect assessments of the databases, thus circumventing the problem that ground truth is not known. It is concluded that with the exception of a few dimensions, the CVSS is quite trustworthy. The databases are relatively consistent, but some are better than others. The expected accuracy of each database for a given dimension can be found by marginalizing confusion matrices. By this measure, NVD is the best and OSVDB is the worst of the assessed databases.

Index Terms—software vulnerability, CVSS, information security, cyber security

1 Introduction

The Common Vulnerability Scoring System (CVSS) is probably the most used system for assessing software vulnerabilities. As such, it is an integral part of much of the security work that goes on globally. In particular, CVSS scores are provided from many data sources, such as the U.S. National Vulnerability Database (NVD).

Many articles are based on data from vulnerability databases such as NVD, aiming e.g. to predict software vulnerabilities [1], evaluate security policies [2], or even create appropriate taxonomies for security [3].

However, the data quality in these databases has received some criticism, and alternative ways to calculate vulnerability scores have been suggested [4], [5], [6] though most often without empirical justification. In the practitioner community, the reliance on CVSS has also been challenged, as new threat and vulnerability management solutions have emerged, aiming for "more granular and intelligent remediation strategies than simplistic severity or Common Vulnerability Scoring System (CVSS)-based approaches" [7]. There are also some evaluations of CVSS improvements, particularly of version 2 over version 1 [8]. This literature thus reports some shortcomings in CVSS, leading to our research question: how trustworthy is CVSS?

More precisely, we wish to estimate the credibility of the data found in five vulnerability databases; NVD, X-Force, OSVDB, CERT-VN, and Cisco IntelliShield Alerts. As a corollary, we can also say something about the CVSS scoring system as such. This analysis would be straightforward if there was a ground truth against which each and every database could be evaluated. However, as the only 'truth' available is the database assessments themselves, instead it may seem like an impossible problem to solve.

- *P. Johnson, R. Lagerström, and M. Ekstedt are with KTH Royal Institute of Technology, SE-100 44 Stockholm, Sweden. E-mail: {pontusj, robertl, mekstedt}@kth.se*
- *U. Franke is with SICS – the Swedish Institute of Computer Science, Isafjordsgatan 22, SE-164 40 Kista, Sweden. E-mail: ulrik.franke@sics.se*

Manuscript received . . .

The beauty of the Bayesian learning method used is that it offers a solution to this seemingly intractable problem. In short, the Bayesian solution is to start from prior beliefs and iteratively update these beliefs with new observations according to Bayes' theorem. Carefully selecting priors so as not to distort the final results, we can thus say something meaningful and understandable about the trustworthiness of the databases, even though there is no ground truth available *a priori*.

Based on the analysis, we conclude that CVSS is a trustworthy and robust scoring system in the sense that most of the databases agree most of the time. That is, the databases are relatively consistent with each other. Also a measure of the discrepancies can be found by marginalizing confusion matrices, representing the expected accuracy of each database for a given dimension. By this measure, NVD is the best and OSVDB is the worst of the assessed databases. Further, we see that the databases are most accurate for the CIA (Confidentiality, Integrity, and Availability) metrics, and least accurate for the metric called access complexity. This confirms some of the criticism that has been directed towards CVSS 2.0.

This kind of knowledge can be of use in many different ways. For academia, it aids the interpretation of research results based on CVSS. For industry, it helps prioritizing security efforts. For tool vendors, it allows more reliable threat analyses.

The remainder of this paper is structured as follows. Section 2 introduces the CVSS base scores, the trustworthiness of which is the key concern of the paper. Some related work is then discussed in Section 3, to put the contribution in proper context. The Bayesian learning method employed is described in Section 4, followed by descriptions of the data sources in Section 5. The main results of the investigation – the trustworthiness assessments of the vulnerability databases with respect to different kinds of vulnerabilities and severities – are then presented in Section 6. The reliability, validity, and usefulness of the results are discussed in Section 7, before Section 8 concludes the paper.

2 Common Vulnerability Scoring System

The Common Vulnerability Scoring System (CVSS) version 1 was first suggested in February 2005 based on research by the National Infrastructure Advisory Council (NIAC), but was early handed over to the Forum of Incident Response and Security Teams (FIRST). Version 2 of CVSS was released in June 2007, and most recently a version 3 was released in June 2015. In this paper we use CVSS v2.0 since most of the 75 241 (February 2016) vulnerabilities in the Common Vulnerabilities and Exposures (CVE) list are scored according to this version.

CVSS is meant to be a standardized and open approach for rating vulnerabilities. For practitioners, a key advantage of using CVSS is that it offers a useful and repeatable approach for ranking the criticality of vulnerabilities [9].

The scoring system is built around three metric groups; Base, Temporal, and Environmental. Base metrics represent the basic features of a vulnerability. These metrics do not change over time or due to specific surroundings or situations. Temporal metrics instead represent the time characteristics of a vulnerability. Finally, the environmental metrics characterize unique user environments.

In this paper we focus on the base metrics since we are aiming at investigating the correctness of the scoring system in general and not for any specific time or place.

The base metric group contains six metrics illustrated in Table 1. Three focus on how the vulnerability is accessed (access vector and complexity) and whether any extra conditions are necessary for exploiting it (authentication). The other three are so called impact metrics focusing on what damages an exploited vulnerability might cause in terms of loss of confidentially, integrity and availability. Each metric is scored based on an ordinal scale of three.

3 Related work

There are a few other analyses that address similar questions, i.e. the quality or usefulness of CVSS. Allodi et al. investigate whether high CVSS scores correspond to the vulnerabilities being exploited more often in the wild [10] [11]. Investigating this using (i) a database of vulnerabilities sold on the black market and (ii) a Symantec database of vulnerabilities encountered in the wild, the authors conclude that CVSS scores show high sensitivity but very low specificity for exploits used in the wild [10]. As a consequence, patching strategies that are based on monitoring vulnerability markets outperform those based on CVSS [11].

Zhang et al. use NVD data to build prediction models of time to next vulnerability (TTNV) [12]. With respect to CVSS, their results are inconclusive, i.e. sometimes the inclusion of CVSS scores improved predictions (e.g. for Microsoft), sometimes not (e.g. for Linux).

Another way to evaluate CVSS is to measure whether there is a correlation between CVSS scores and time to compromise. An empirical investigation using real compromises from a cyber defense exercise shows no strong correlations of this kind [13]. Still, the CVSS approach of composing several metrics gets some vindication from this study, as time to compromise exhibits more correlation to metrics composed from more data than simpler metrics just focusing on the most severe vulnerability (i.e. weakest link). The CVSS "exploitability" metric has also been outperformed by a machine learning approach at the task of representing the likelihood that a vulnerability is exploited [14]. However, in each of these cases, it should be noted that CVSS is *not* constructed specifically to measure any of the quantities (prevalence of vulnerability exploitation *in vivo*, time to compromise, or availability of exploits) investigated. Instead, CVSS aims to offer a single relevant high level score, as described in Section 2.

Holm and Afridi study the agreement between CVSS base scores and expert judgments of 3 040 vulnerabilities, finding a significant but small disagreement, though the variance was considerable [15]. While the research question of Holm and Afridi is similar to ours in the sense that they evaluate CVSS scoring, this is addressed on different levels of granularity: Holm and Afridi look at the compound base score that can be calculated as a one-dimensional measure of all the six base metrics illustrated in Table 1. We, on the other hand, are not concerned with this compound score, but only address the trustworthiness of the six base metrics.

4 Method

In order to estimate the accuracy of the CVSS scoring performed by the teams of the five examined vulnerability databases we have devised a statistical model to perform the data analysis. The statistical model has been populated with data about individual vulnerabilities (CVEs), retrieved from the databases' respective web interfaces.

For the data analysis we use Bayesian statistics. There is an ongoing debate between the frequentist and the Bayesian schools over how to interpret and apply statistics. We will not describe this debate here, nor give detailed arguments for using the Bayesian paradigm. We summarize that the frequentist interpretation has been dominant historically, but today both schools have strong support and both are widely used in many scientific areas. A primary motivation for choosing the Bayesian approach is that it lends itself to an interpretation of uncertainty useful for our purposes; the probability of an event occurring or of a statement being true represents an opinion of how likely it is that the event occurs or that the statement is true. With more observations, opinions are rationally updated by Bayesian inference rules [16] [17].

The model employed in the present research is an hierarchic Bayesian model. Such models are particularly suitable when populations are expected to differ in important ways, while still sharing some common properties. For instance, [18] uses an example of the batting performance of a softball player as an example suitable for hierarchic Bayesian analysis. In this example it is possible to capture the fact that the probability of a player getting a hit is dependent on the properties particular for a certain game, as well as the properties that the batter maintains from game to game. With respect to the vulnerability databases, we hypothesize that the different scoring teams have some commonalities, such as public vulnerability information on which to base the scoring, while differing in other aspects, such as the scorers' background knowledge.

This section first describes the statistical model, including its structure and parameter dependencies, before the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2016.2644614, IEEE Transactions on Dependable and Secure Computing

TABLE 1
Base metric scores in CVSS. Descriptions are taken from https://www.first.org/cvss/v2/guide

Metric	Values	Description
Access vector	Local (L)	A vulnerability exploitable with only local access requires the attacker to have either physical access to the vulnerable system or a local account.
	Adjacent Network (A)	A vulnerability exploitable with adjacent network access requires the attacker to have access to either the broadcast or collision domain of the vulnerable software.
	Network (N)	A vulnerability exploitable with network access means the vulnerable software is bound to the network stack and the attacker does not require local network access or local access. Such a vulnerability is often termed "remotely exploitable".
Access complexity	High (H)	Specialized access conditions exist.
	Medium (M)	The access conditions are somewhat specialized.
	Low (L)	Specialized access conditions or extenuating circumstances do not exist.
Authentication	Multiple (M)	Exploiting the vulnerability requires that the attacker authenticate two or more times, even if the same credentials are used each time.
	Single (S)	The vulnerability requires an attacker to be logged into the system.
	None(N)	Authentication is not required to exploit the vulnerability.
Confidentiality	None(N)	There is no impact to the confidentiality of the system.
	Partial (P)	There is considerable informational disclosure. Access to some system files is possible, but the attacker does not have control over what is obtained, or the scope of the loss is constrained.
	Complete (C)	There is total information disclosure, resulting in all system files being revealed.
Integrity	None(N)	There is no impact to the integrity of the system.
	Partial (P)	Modification of some system files or information is possible, but the attacker does not have control over what can be modified, or the scope of what the attacker can affect is limited.
	Complete (C)	There is a total compromise of system integrity. There is a complete loss of system protection, resulting in the entire system being compromised. The attacker is able to modify any files on the target system.
Availability	None(N)	There is no impact to the availability of the system.
	Partial (P)	There is reduced performance or interruptions in resource availability.
	Complete (C)	There is a total shutdown of the affected resource. The attacker can render the resource completely unavailable.

initial state of the model is described in terms of prior distributions. The third subsection describes how the model parameters are estimated. Finally, the practical implementation of the model is briefly summarized.

4.1 The statistical model

The statistical model is based on CVSS scores of individual vulnerabilities. For each vulnerability j (e.g. CVE-2009-1253), we assume that there exists a true value for each CVSS scoring dimension s. (In Section 7, we critically discuss this assumption.) $M^j = \{\mu^j_{av}, \mu^j_{ac}, \mu^j_{au}, \mu^j_c, \mu^j_i, \mu^j_a\}$ denotes the true values of the CVSS scoring dimensions *access vector*, *access complexity*, *authentication*, *confidentiality impact*, *integrity impact*, and *availability impact* of vulnerability j. Although the labels differ, the range of values for each dimension is always three, $\mu_s \in \{\nu^1_s, \nu^2_s, \nu^3_s\}$. For instance, the authentication dimension may assume the values `none`, `single instance`, and `multiple instances`. Cf. Table 1 for all CVSS scoring dimensions.

For a randomly selected vulnerability we assume that its value in a chosen CVSS scoring dimension is given by a categorical probability distribution, $P(\mu_s = \nu^1_s) = \psi^1_s$, $P(\mu_s = \nu^2_s) = \psi^2_s$, and $P(\mu_s = \nu^3_s) = \psi^3_s$ where $\psi^1_s + \psi^2_s + \psi^3_s = 1$. The set of all distributions is concisely referred to as $\Psi = \{\psi_{av}, \psi_{ac}, \psi_{au}, \psi_c, \psi_i, \psi_a\}$.

Each respective vulnerability database management team may for a given vulnerability j choose to assign a CVSS score, $\chi^j_s \in X$, where $X = \{\chi_{av}, \chi_{ac}, \chi_{au}, \chi_c, \chi_i, \chi_a\}$ corresponds to the CVSS scoring dimensions. However, these assignments are not necessarily correct. Databases may be associated with varying sizes of systematic errors, e.g. a bias overemphasizing the gravity of vulnerabilities, as well as with random errors, e.g. the performance of individual members of the scoring teams. To represent these inaccuracies, for each database and CVSS scoring dimension, we introduce a 3×3 confusion matrix

$$\Pi^s = \begin{bmatrix} \pi^s_{11} & \pi^s_{12} & \pi^s_{13} \\ \pi^s_{21} & \pi^s_{22} & \pi^s_{23} \\ \pi^s_{31} & \pi^s_{32} & \pi^s_{33} \end{bmatrix} \quad (1)$$

For instance, for the CVSS dimension *confidentiality impact*, the confusion matrix looks like

$$\Pi^c = \begin{bmatrix} \pi^c_{cc} & \pi^c_{cp} & \pi^c_{cn} \\ \pi^c_{pc} & \pi^c_{pp} & \pi^c_{pn} \\ \pi^c_{nc} & \pi^c_{np} & \pi^c_{nn} \end{bmatrix} \quad (2)$$

where π^c_{cc}, e.g., denotes the probability that the considered database will correctly assign a random vulnerability of the actual confidentiality impact score `complete` with that same value, while π^c_{cp} and π^c_{cn} represent the probabilities that the database incorrectly would assign such a complete-impact vulnerability the values `partial` or `none`, respectively.

Finally, the probability that a given database team assigns a specific value of a certain CVSS scoring dimension to a randomly selected vulnerability depends on both the true value and the accuracy of the assignments following a categorical distribution:

$$P(\chi^j_s = \nu^y_s) = \sum_{x \in \{1,2,3\}} \mu^x_s \pi^s_{xy}. \quad (3)$$

Of course, we do not know the values of the variables ψ_s and π^s. Rather these are the values we wish to estimate. In particular, π^s for all CVSS dimensions will provide an exhaustive account of the degree to which we can trust the CVSS scoring in each database. The statistical model, visualized in Fig. 1 is a hierarchic Bayesian model with two layers, where the first stage captures the scores of the individual vulnerabilities and the second stage captures

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2016.2644614, IEEE Transactions on Dependable and Secure Computing

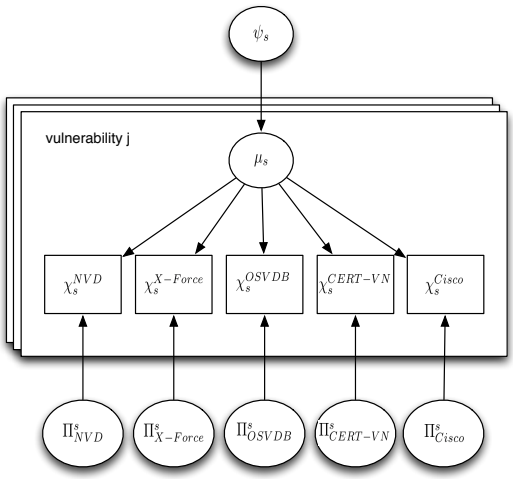

Fig. 1. The model depicted as a directed acyclic graph, representing the parameters and their dependencies as described in the text. Rectangles denote observed data and circles denote unknowns. The stacked sheets denote the repetitive observations over the same variable, in our case each vulnerability j. Every node in the figure contains all the respective CVSS scoring dimensions s. I.e. the fully sketched out model is thus one with six identical model structures, one for every scoring dimension s.

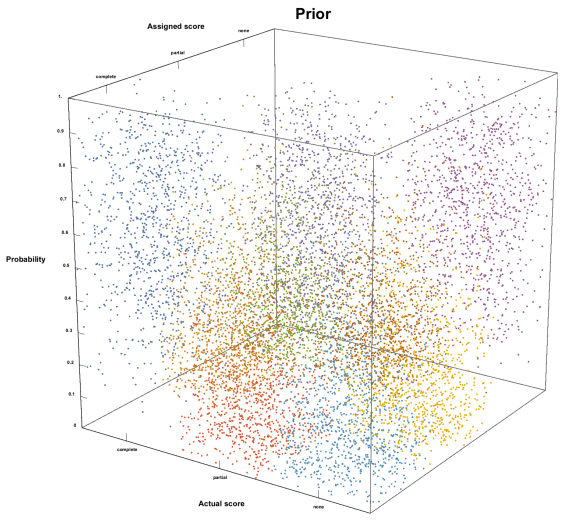

Fig. 2. The prior distributions of belief over the confusion matrices. A confusion matrix specifies the relation between *assigned* score and *actual* score. For an ideal scorer all the dots should be on the diagonal, just as in a numerical confusion matrix (cf. Appendix A). Different colors are used to distinguish each of the nine positions in the 3×3 state space. Due to the uninformative nature of the priors illustrated, the dots may be difficult to interpret, looking like a random cloud filling the cube. Therefore, it is useful to contrast this figure with e.g. the lower (CIA) plots in Fig. 3 (NVD scores) which are quite close to representing a perfect scorer. The vertical axis represents the *probability distribution* of a certain scoring, thus reflecting the uncertainty of the analysis, information which is unavailable in the numerical confusion matrices of Appendix A. A fully correct and 100% certain assessment would be depicted as a cube with all the samples located on the top plane ($y = 1$) in the three diagonal columns. In this figure, however, the dots are samples from the prior distributions Dirichlet(3,1,1), Dirichlet(1,3,1) and Dirichlet(1,1,3), representing beliefs corresponding to the observation of three correct scores (e.g. scoring `complete` when the actual value was `complete`, and one incorrect score for `partial` and `none` respectively). An equal number of samples, sufficient to illustrate sample variance, is shown for each of the three dimensions. As can be seen, the correct scores, located along the cube diagonal, have somewhat higher probability than incorrect scorings, but the "density of the clouds" is low, i.e. variance is high, for all combinations, reflecting the uncertainty.

the accuracies of the vulnerability scoring as well as the probability distribution for vulnerabilities in general. As is common when modelling measurement error, the measurement error is assumed independent of the truth parameter.

4.2 Prior distributions

In order to solve Bayesian models it is necessary to include prior distributions of variables, i.e. how we believed the variables were distributed before the study was performed. The issue of priors has been discussed at length within the statistical community, as this seems to introduce a subjective element into every statistical result. In order to limit this subjective influence, it is common to employ so called *uninformative priors*, i.e. priors that represent *a priori* ignorance of the considered distributions. Taking this as our starting point, we choose a ternary Dirichlet distribution as the prior for ψ_s and π^s. One common set of parameter values to represent uninformedness in the Dirichlet distribution is $Dir(1,1,1)$, which is tantamount to a uniform prior over the probability space. The $Dir(1,1,1)$ prior can be interpreted as a belief corresponding to a single observation of each of the three parameter values. It will therefore have little impact on the posterior belief if the observations in the study at hand are significantly more numerous than three.

We also need priors for the confusion matrices. The first impulse would perhaps be to use $Dir(1,1,1)$ also for π^s. However, this will create a model with multiple solutions to any data set. E.g., it will not be possible to differentiate between (i) a model where the hidden, actual value of the *integrity impact* of a vulnerability is `complete`, and the databases represent this correctly (so that π^s_{cc} is near unity), and (ii) a model where the databases are consistently misinterpreting `complete` as `none` for that same vulnerability (so that instead π^s_{cn} is near unity). This second model would not be incorrect in the sense that it is mistakenly identifying a bad vulnerability scorer, it would just be a scorer that had confused the labels syntactically. To avoid the second, undesired outcome of the parameter learning process, we use the priors $\Pi^s = [Dir(3,1,1)^T, Dir(1,3,1)^T, Dir(1,1,3)^T]$. This distribution is illustrated graphically in Fig. 2.

This represents an initial belief that the databases' scoring teams are not completely random in their work and that they are at least somewhat more right than wrong in their syntactic label mapping. The priors are, however, still weak, as the subsequent observations are in the thousands. The graphical depiction of the priors in Fig. 2 should be contrasted to the posterior distributions, illustrated in in Figs. 3-7 in the result section below.

4.3 Parameter estimation

Given the model and the prior distributions detailed in the previous subsections, we are now set to estimate the unknown parameters – in particular the databases' scoring accuracy as summarized in their respective confusion matrices – based on available data, i.e. the set of CVSS-scored vulnerabilities in the five databases.

4.3.1 Bayesian learning

At first sight, it might appear difficult to estimate the scoring accuracy when the actual, true, CVSS values are not known.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2016.2644614, IEEE Transactions on Dependable and Secure Computing

After all, we only have the potentially inaccurate scores in the five databases. However, a Bayesian approach is to a large extent able to handle this apparent problem. In essence, the Bayesian approach to solving the model is to iteratively update prior beliefs with observations thus generating posterior beliefs. Bayes' rule states that posterior belief is proportional to ($\propto$) the prior belief times the likelihood:

$$P(Model|Data) \propto P(Model) \times P(Data|Model) \quad (4)$$

We seek to estimate the model parameters Π, Ψ, and M, given the CVSS scores, X, provided by the databases, i.e. we want to solve $p(\Pi, \Psi, M|X)$. In our case Bayes' rule thus corresponds to:

$$P(\Pi, \Psi, M|X) \propto P(\Pi, \Psi, M) \times P(X|\Pi, \Psi, M) \quad (5)$$

An alternative way to intuitively understand the concept of solving the model is to employ the frequentist approach to parameter estimation, namely maximum likelihood estimation. The goal of such an exercise is to find the model parameters that maximize the probability of seeing the actual data. Whichever approach is employed, any correlations between the databases need to be captured in the marginal probabilities of the actual scores, M, while a faithful model of the remaining local scoring bias requires fitting the elements of the confusion matrices, Π, to the observed data.

4.3.2 Model fitting

In practice, Bayesian model fitting problems are not solved analytically, but numerically using Markov Chain Monte Carlo methods. We have employed the Gibbs sampling method, which is a good choice for models where the complete joint posterior distributions cannot be analytically determined nor directly sampled, but all individual conditional distributions can be determined as well as directly sampled [17].

Explained briefly, the Gibbs sampler uses the full Bayesian model and produces a sample for each model parameter, given assumed known values of all the other model parameters. The algorithm traverses the full model for every iteration in a specified number of iterations. Cowels [18] neatly describes this in the following syntax: assume a Bayesian model with n parameters with unknown values α_1, α_2, ..., α_n. The model is sampled several times by the Gibbs sampler, indicated by a superscript where the initial values are denoted $\alpha_1^{(0)}$, $\alpha_2^{(0)}$, ..., $\alpha_n^{(0)}$. For any iteration k the Gibbs sampler draws a value for each parameter from the full conditional distribution, given all the other parameters' most recent values:

- Draw $\alpha_1^{(k)}$ from $p(\alpha_1|\alpha_2^{(k-1)}, \alpha_3^{(k-1)}, ..., \alpha_n^{(k-1)})$
- Draw $\alpha_2^{(k)}$ from $p(\alpha_2|\alpha_1^{(k)}, \alpha_3^{(k-1)}, ..., \alpha_n^{(k-1)})$
- ...
- Draw $\alpha_n^{(k)}$ from $p(\alpha_n|\alpha_1^{(k)}, \alpha_2^{(k)}, ..., \alpha_{n-1}^{(k)})$

Every single step thus becomes simple, however finding the full conditional distribution typically is not. It is possible to analytically derive these functions and sometimes they follow well-known distributions, but often not. Fortunately, Gibbs sampling software[1] is today readily available that takes care of this task. If the software does not find a known distribution it employs an algorithm for sampling non-standard distributions. The initial values $\alpha^{(0)}$ can be derived from the prior distributions (or by other means, if preferred). Also this task can be left to the software. The Gibbs sampler thus numerically reconstructs the distributions of all the parameters in the model one point at the time. However, since the algorithm is initiated with random numbers, the first samples may not be representative of the posterior distributions. This is referred to as the "burn-in" period and the common practice is to discard those samples. Exactly how long the burn-in period should be varies with the model and the data at hand.

Gibbs sampling is not always an appropriate model fitting method. If the distributions of the model parameters have extreme probability densities, e.g. located to small spaces, the Gibbs sampler will not be able to "break out" of such spaces when reconstructing the model parameter distributions since every parameter sampling is conditioned on the probability densities of all the other parameters. In such cases the Metropolis-Hastings sampling method is typically used instead. In our case, the Dirichlet distribution is used because it is the conjugate prior of the multinomial distribution, the use of which follows from the problem of finding confusion matrices. With the priors used, the Dirichlet distribution is smooth and well-behaved, thus allowing us to use Gibbs sampling. The interested reader is referred to e.g. [16] [17] [18] [19] for a more detailed description of Bayesian model fitting.

In summary, to come to a valid solution using Gibbs sampling, four steps need to be fulfilled [19]. Firstly, initial values for all parameters must be provided. In our case we provided the priors and used the built in feature of JAGS[2] for setting the initial values [20]. Secondly, the full conditional distributions of the models need to be derived. JAGS also automates this task. Thirdly, the output of the sampling must be monitored for its burn-in period. This was done and the 250 first samples were thrown away. Fourthly, statistics of the unobserved models parameters should be calculated. In our case this statistic is captured in the confusion matrices Π and the "density of the sample cloud" for the different CVSS scoring dimensions.

4.4 Implementation

The language R [21] was used for processing the vulnerability data from the various databases. The NVD database is available for download in XML format, which makes it easy to import into R. Information from the remaining databases, however, was not as easily accessible. Therefore, programs for exploring these databases online were developed in R, downloading page content and then parsing, primarily using regular expressions. As mentioned, Just Another Gibbs Sampler (JAGS) [20] was used for formalizing the statistical model and for executing the Gibbs sampling. The rjags package [22] was employed to interface R with JAGS. As simulations required significant computational

1. In this work we have used the Just Another Gibbs Sampler (JAGS) software, https://sourceforge.net/projects/mcmc-jags

2. From the manual we read "If initial values are not supplied by the user, then each parameter chooses its own initial value based on the values of its parents. The initial value is chosen to be a 'typical value' from the prior distribution. The exact meaning of 'typical value' depends on the distribution of the stochastic node, but is usually the mean, median, or mode."

resources, the Amazon Elastic Cloud Web Service platform[3] was employed to increase the processing capacity. Wolfram Mathematica [23] was employed to visualize the three-dimensional confusion matrices.

5 Data

NVD, X-Force, OSVDB, CERT-VN, and Cisco are some of the biggest databases with independent CVSS scoring. In this section these databases are introduced, along with what is known about their CVSS scoring principles and possible interdependencies. It should be noted that the databases differ with respect to the vulnerability data they offer, and due to legacy issues do not always offer the same data for all vulnerabilities. The 'least common denominator' defining the selection criteria for inclusion in the study is that a vulnerability; (i) has CVSS scores in the six dimensions explained in Table 1, (ii) has an assigned CVE number, and (iii) the scores are not credited to another database (for example, OSVDB displays multiple scores, credited with sources such as NVD). The data used in the study is the six CVSS base scores, as explained in Fig. 1, though some databases contain additional information.

5.1 NVD

The National Vulnerability Database (NVD) is a U.S. government repository of vulnerability data. It contains several kinds of data, one of which is the database of software flaws identified by CVE numbers (Common Vulnerabilities and Exposures). As of February 2016, NVD contains 75 241 CVE vulnerabilities (not all of which scored by CVSS 2.0).[4]

NVD supports the CVSS v.2 standard for all CVE vulnerabilities, providing base scores. NVD expresses its willingness to collaborate with the security community on CVSS scoring, and the public is invited to contribute additional information or corrections. Nevertheless, there is every reason to believe that most NVD scoring is done independently from other databases.[5] 39 338 independent NVD CVSS 2.0 scores were used.

5.2 X-Force

X-Force is the IBM security brand, focusing on threat intelligence. According to its website, its database contains more than 88 000 computer security vulnerabilities.

The X-Force Exchange does not describe in detail how its CVSS assessments are made. In the X-Force Exchange FAQ, threat information in general is described as emanating from; (i) IBM internal infrastructure and databases, (ii) open-source content, and (iii) third-party partnerships. On vulnerabilities specifically, the FAQ merely states that the X-Force Database is one of the oldest, publicly available vulnerability databases in the world.[6] 29 034 independent X-Force CVSS scores were used.

5.3 OSVDB

The Open Source Vulnerability DataBase (OSVDB) describes itself as an independent and open sourced web-based vulnerability database, originally announced in 2002, and supporting CVSS 2.0 scoring since October 2009. It contains more than 100 000 entries, and describes over 99% of its contents and contributions as coming from a few volunteers and employees hired by its parent organization, the Open Security Foundation (OSF), or Risk Based Security, its primary sponsor.[7] On April 5, 2016, after our data collection and analysis was completed, OSVDB announced that the database will be shut down.[8]

In the OSVDB FAQ, the vulnerability information is described as coming from; (i) security mailing lists, (ii) exploit aggregation sites, (iii) vendor websites, and (iv) a *lot more* (emphasis in original). To some extent, the information is crowdsourced, as anyone can submit a vulnerability to a moderator for inclusion, and it is also possible to submit updates or requested changes.[9]

When OSVDB support for CVSS was announced in October 2009, it was made clear that the scores displayed were "mostly as calculated by the National Vulnerability Database (NVD)". Importantly, OSVDB also offered its users an interface to disagree with NVD scores.[10]

However, as of now, it seems that the OSVDB CVSS scoring is much more independent of NVD. First, numerous vulnerabilities differ in their NVD and OSVDB scores, and OSVDB scores are often older than NVD scores. Second, some vulnerabilities that have no NVD score nevertheless have an OSVDB score. Third, there are vulnerabilities where two identical CVSS scores (not only in terms of the final value, but of all the six constituents) are credited and dated differently – one to NVD, one to OSVDB. Thus, we believe that the CVSS scores credited to OSVDB in the database are always the results of independent OSVDB scoring efforts. These are the OSVDB CVSS scores used in our analysis.

OSVDB also contains a number of vulnerabilities with a CVSS score credited to NVD, but no CVSS score credited to OSVDB. Many of these date from before October 2009 when OSVDB started supporting CVSS scoring. Of course, scores found in OSVDB but credited to NVD are excluded from our analysis, as they do not represent independent OSVDB scoring. 489 independent OSVDB CVSS scores were used.

5.4 CERT-VN

The CERT Division of the Software Engineering Institute (SEI) at Carnegie Mellon University hosts the Vulnerability Notes Database (CERT-VN), with information about software vulnerabilities. The entries are described as emanating from "private coordination and disclosure efforts".

CERT-VN has offered CVSS scores – base, environmental, and temporal – since March 27, 2012. For CVSS scoring, CERT-VN report that they are in close contact with NVD and work with them to synchronize the way CVSS metrics are scored: "The lines of communication are open so that we can resolve any discrepancies quickly and easily." The general public is also invited to offer dissenting opinions on CVSS scores via e-mail.[11]

3. http://aws.amazon.com/ec2/
4. https://nvd.nist.gov/
5. https://nvd.nist.gov/cvss.cfm
6. https://exchange.xforce.ibmcloud.com/faq
7. http://osvdb.org/about
8. https://blog.osvdb.org/2016/04/05/osvdb-fin/
9. http://osvdb.org/faq
10. https://blog.osvdb.org/2009/10/06/osvdb-now-supports-cvssv2-scoring/
11. https://insights.sei.cmu.edu/cert/2012/04/-vulnerability-severity-using-cvss.html

Thus, while it cannot be ruled out that CERT-VN and NVD scoring influence each other, it is clear that CERT-VN performs independent CVSS scoring. There are many examples where vulnerabilities differ in their CVSS scores as assigned by CERT-VN and NVD, where CERT-VN scores are older than NVD scores, and where CERT-VN have scored vulnerabilities not found in NVD (or at least not assigned a CVE). 309 independent CERT-VN CVSS scores were used.

5.5 Cisco IntelliShield Alerts

The Cisco Product Security Incident Response Team (PSIRT) provides vulnerability information including CVSS scoring.[12] Vulnerability information is made publicly available under the heading Cisco IntelliShield Alerts.

Cisco uses CVSS 2.0 to evaluate vulnerabilities in Cisco products, always offering the base score and sometimes also the temporal score. For third-party software, Cisco typically uses the CVSS score provided by the software creator, but reserves the right to adjust the CVSS score "to reflect the impact to Cisco products". The Cisco PSIRT cooperates with other CERTs to coordinate disclosure of vulnerabliities that may affect several vendors.[13]

There is every reason to believe that the Cisco CVSS scoring is an independent effort. Comparing Cisco scores to NVD scores, there are many examples where they differ, and also examples where Cisco has scored vulnerabilities not scored by NVD. 10 430 independent Cisco CVSS scores were used.

6 Results

The results of our analyses are presented in four subsections. In subsection 6.1 we answer the question: How good is each database at predicting the true values of a vulnerability in the six CVSS base score dimensions? Subsection 6.2 deals with: How well do the databases perform compared to each other? Subsection 6.3 focuses on: How certain are we of a given CVSS score? And finally subsection 6.4: How likely is each CVSS score?

6.1 Confusion matrices

In the following, the main result is reported: How good is each database at predicting the true values of a vulnerability in the six CVSS base score dimensions? The answer is given through confusion matrices, i.e. graphical representations of how predicted classes fare against the actual classes. (In our case, of course, the actual classes are hidden variables inferred from the statistical model, as explained in Section 4.) Standard numerical confusion matrices for each of the databases, with percentages in each cell, are given in Appendix A. However, the fullest appreciation of our results is conveyed through Figs. 3-7, where the confusion matrices are depicted with a third dimension absent in the appendix, viz. the probability distribution of the result.

Looking, for instance, at the very first (upper left) of the six cubes depicted in Fig. 3, we see the performance of NVD when classifying vulnerabilities in the Access vector dimension. The thin dark blue point cloud in the upper left of the diagram represents the probability that a vulnerability that is indeed exploitable over the network has been classed as such by NVD. The fact that this cloud is located close to unity means that NVD in general does a good job at this classification. The fact that the cloud is thin means that we are certain of this verdict. The network exploitable vulnerabilities that have been erroneously classified as exploitable from adjacent (orange cloud) or local (green cloud) locations are – as expected – also relatively thin and located close to zero. Looking at vulnerabilities that are indeed exploitable from adjacent locations, we find clouds that are much more dispersed. This corresponds to greater uncertainty in the verdict. Comparing to the numerical representations of Appendix A, the benefit of the graphical representation is clear – the lower value for adjacently exploitable vulnerabilities is evident in the appendix, but not the uncertainty.

Looking at Fig. 3, representing NVD, it is clear that the database in general performs well. This is particularly true for the CIA dimensions, where we see consistently thin clouds in the upper range of the scale along the diagonals representing correct classifications. For access complexity, a poorer result is seen when it comes to classifying vulnerabilities with low access complexity as such. For authentication, we see some really dispersed point clouds when it comes to classifying vulnerabilities requiring multiple authentication as such, representing great uncertainty in this result.

Turning to Fig. 4, representing X-Force, we can again conclude that this database in general performs well. Compared to NVD, there are many similarities, but also differences. For instance, in the CIA dimensions, X-Force does a good job in correctly classifying vulnerabilities with partial or no impact as such, but relatively often misclassifies vulnerabilities with complete impact as having only partial. Similar to NVD, there is great uncertainty when it comes to classifying vulnerabilities requiring multiple authentication as such, but unlike NVD, X-Force is strikingly poor at classifying vulnerabilities requiring single authentication, often classifying them as requiring none.

Looking at OSVDB in Fig. 5, yet other patterns are seen. Again, the CIA dimensions are pretty good, but OSVDB still relatively often misclassifies vulnerabilities with partial impact as having complete. The classification of vulnerabilities exploitable from an adjacent location is both poor and laden with uncertainty. From a methodological point of view, however, the most interesting feature is the access complexity graph. The confusion matrix is 'inverted' – the clouds towards the upper end of the scale are located along the wrong diagonal. In a sense, this matrix says that OSVDB is pretty good at consistently being wrong about access complexity. This phenomenon is related to the priors, and is further discussed in Section 7.2.

The CERT-VN matrices illustrated in Fig. 6 exhibit the weakest classifications of the CIA dimensions – the point clouds are consistently lower than for the other databases, and this is a relatively certain verdict. The certainty of the assessments for access vector, access complexity, and authentication, on the other hand, is relatively low. One result that is certain, however, is that CERT-VN fares well when classifying vulnerabilities that need no authentication.

The results of Cisco, illustrated in Fig. 7, are more certain.

12. https://tools.cisco.com/security/center/viewAllSearch.x
13. http://www.cisco.com/c/en/us/about/security-center/security-vulnerability-policy.html#asr

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2016.2644614, IEEE Transactions on Dependable and Secure Computing

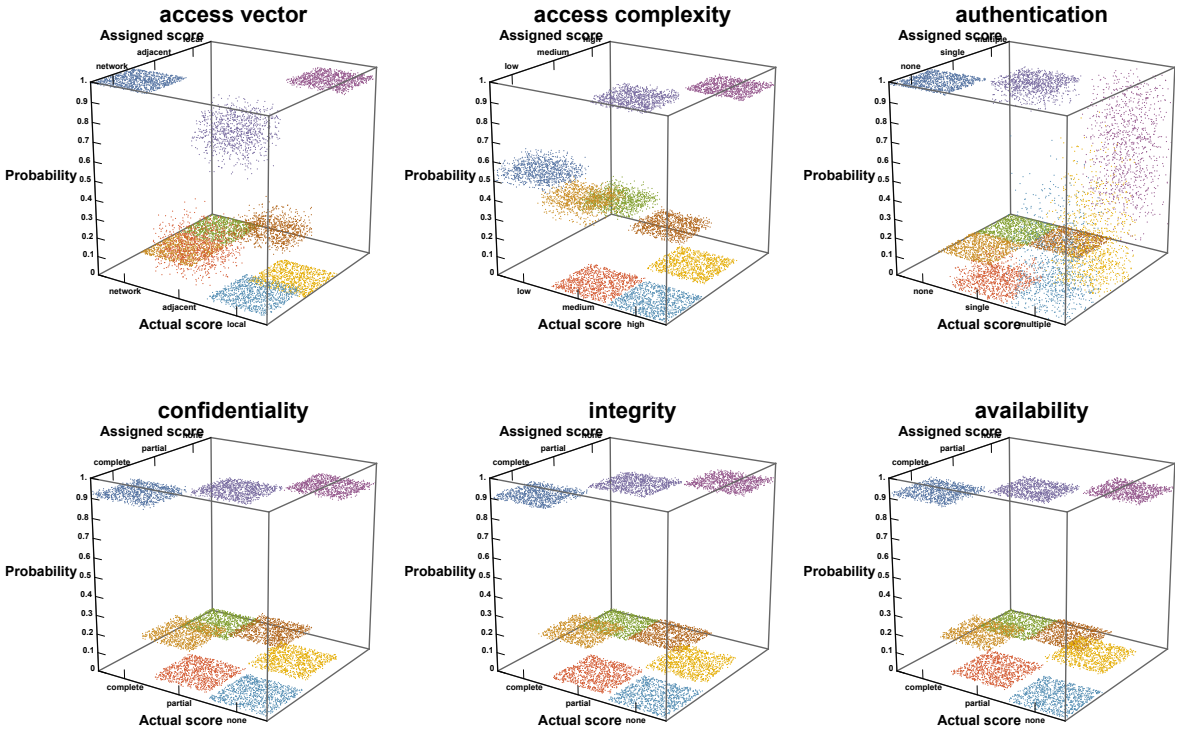

Fig. 3. The NVD confusion matrices.

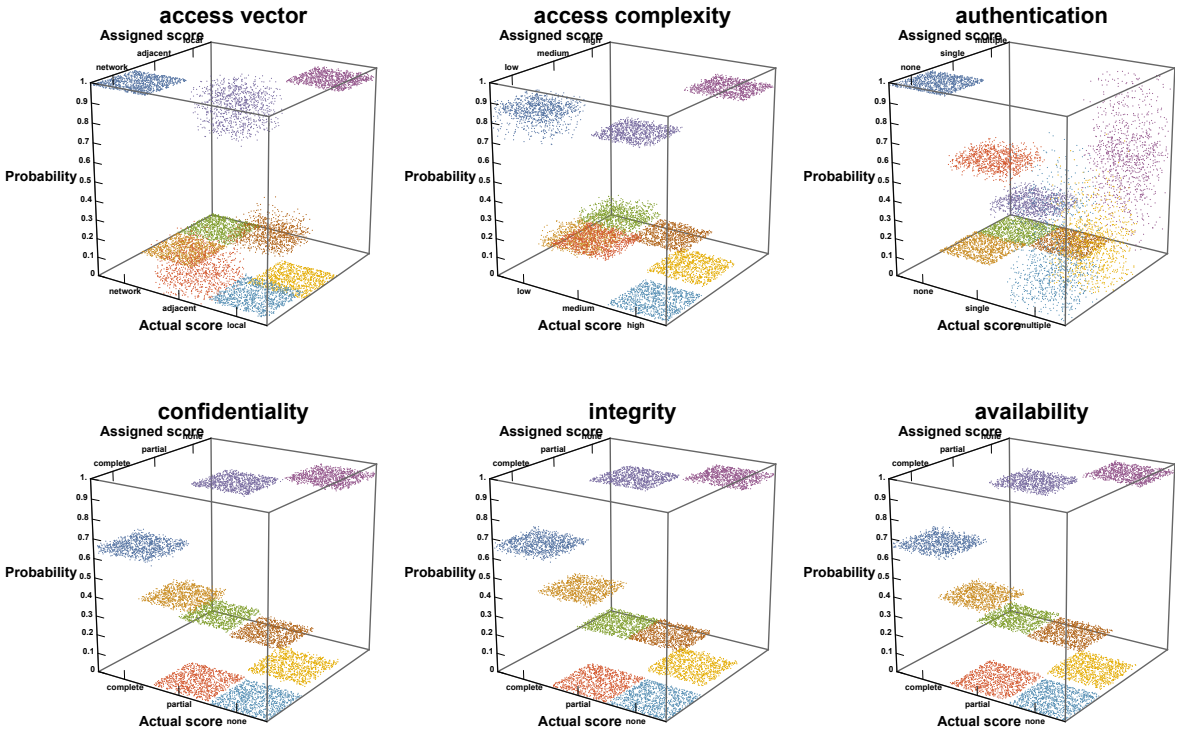

Fig. 4. The X-Force confusion matrices.

Cisco generally fares well in the CIA dimensions, though not as well as NVD. In all the CIA dimensions, there is a consistent problem with misclassification of vulnerabilities with partial impact as having complete (or, to a lesser extent, none), similar to OSVDB. A problem with classifying vulnerabilities with low access complexity as such reminds of NVD, as does the great uncertainty in the result regarding classification of vulnerabilities requiring multiple authen-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2016.2644614, IEEE Transactions on Dependable and Secure Computing

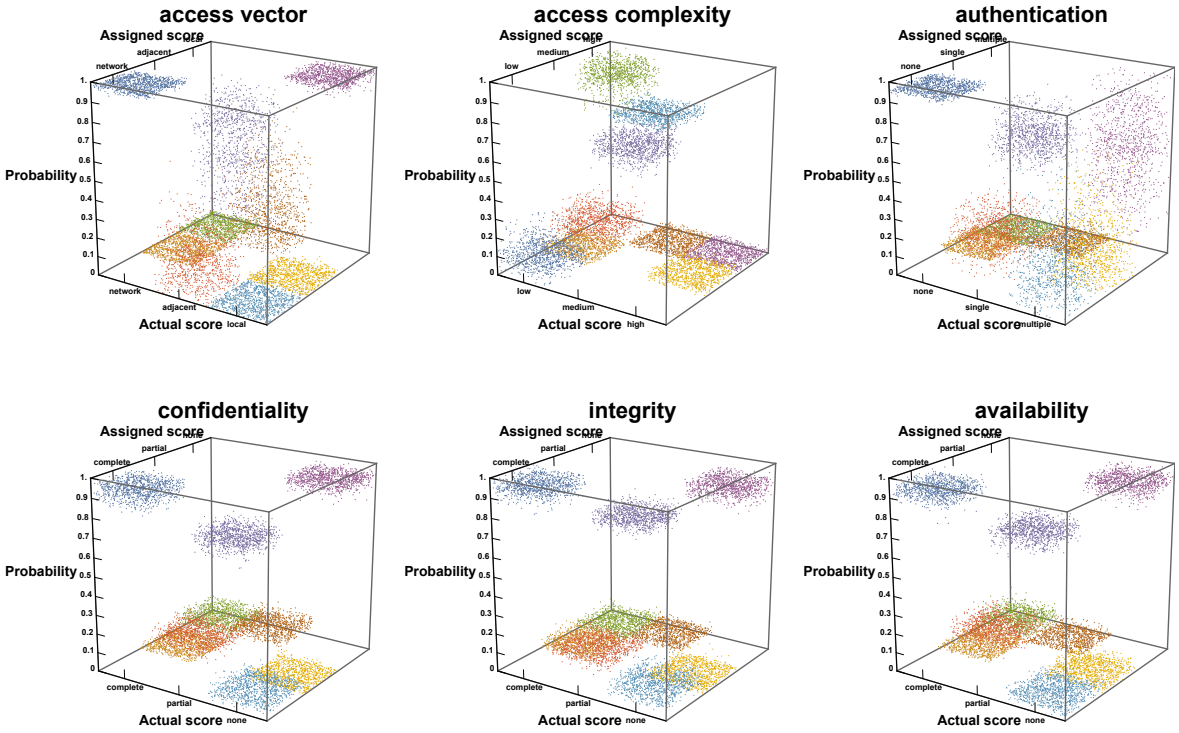

Fig. 5. The OSVDB confusion matrices.

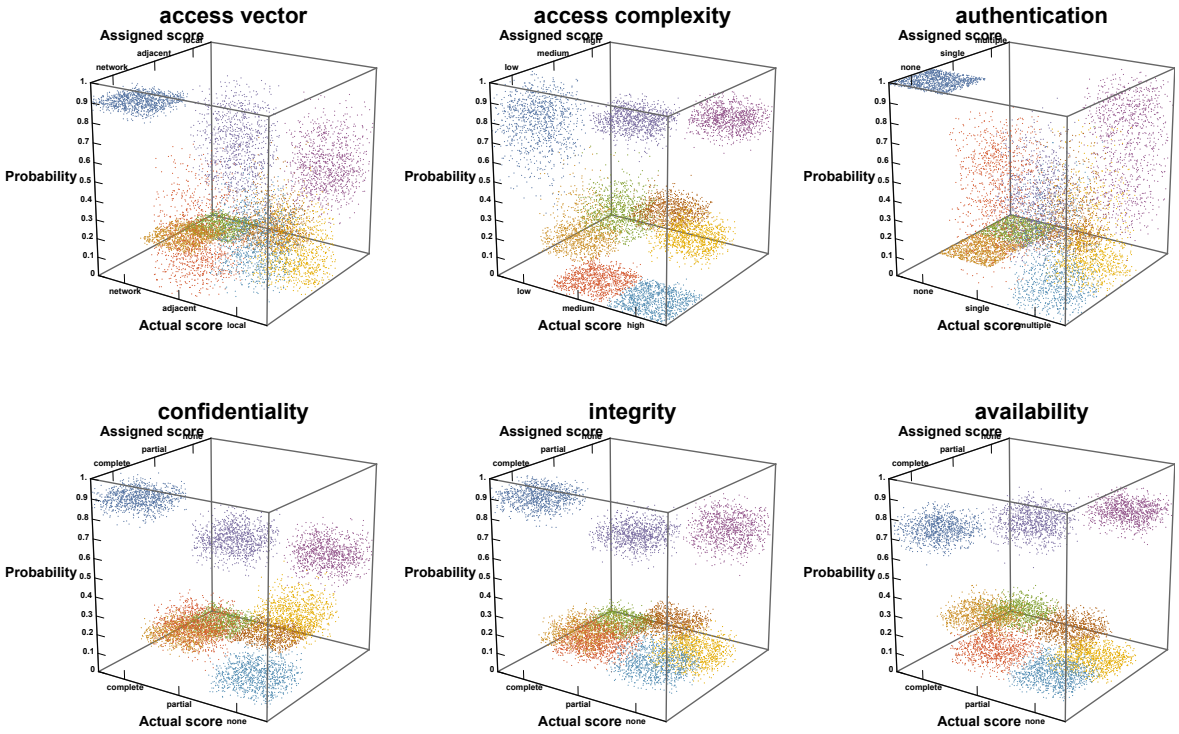

Fig. 6. The CERT-VN confusion matrices.

tication. The results for access complexity assessment are somewhat poorer than those of NVD and X-Force.

6.2 Marginalized accuracy

Each confusion matrix can be marginalized to a single number, representing the expected accuracy of each database for a given dimension. Such marginalization can yield insights

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2016.2644614, IEEE Transactions on Dependable and Secure Computing

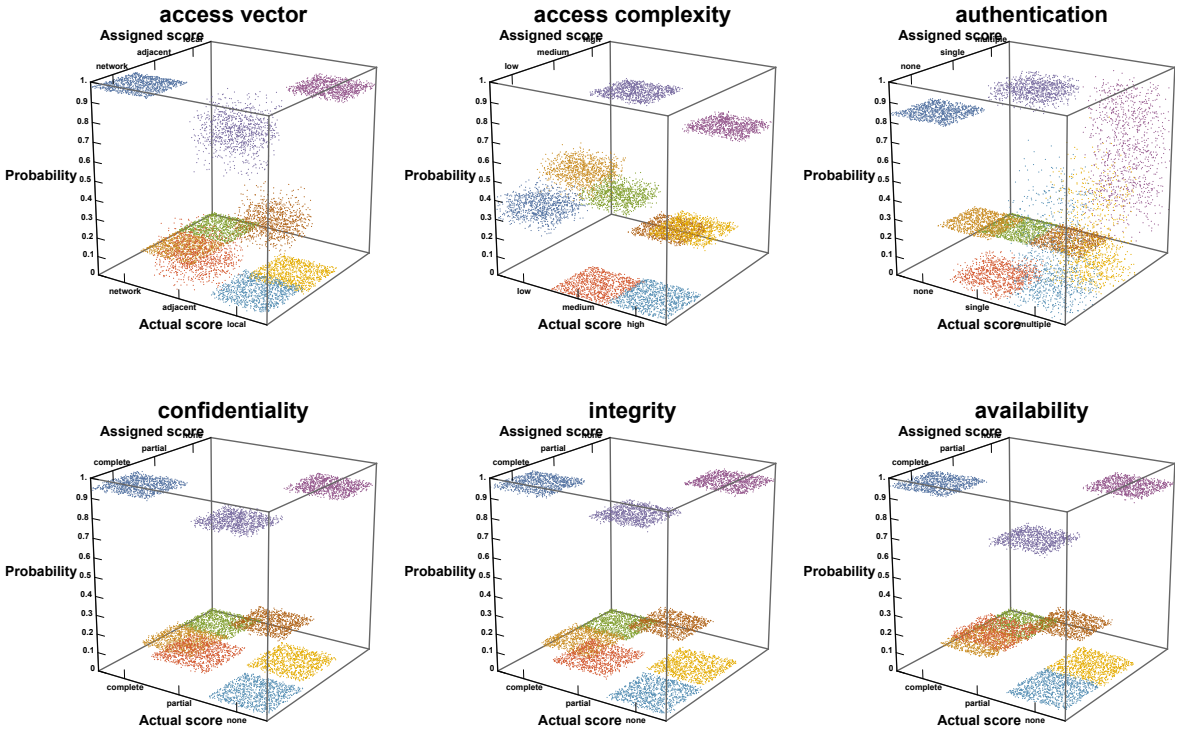

Fig. 7. The Cisco confusion matrices.

TABLE 2
Marginalized accuracies of the databases. Note that the averages are weighted through the marginalization.

Metric	NVD	X-Force	OSVDB	CERT-VN	Cisco
Access vector	99%	98%	97%	85%	97%
Access compl.	88%	84%	24%	76%	76%
Authentication	99%	95%	95%	96%	84%
Confidentiality	90%	88%	81%	67%	84%
Integrity	92%	90%	84%	73%	85%
Availability	90%	88%	83%	75%	81%
Average	**93%**	**91%**	**77%**	**79%**	**85%**

not readily gained from the graphs in Figs. 3-7. For instance, looking at the performance of NVD and X-Force at access complexity classification, one might get the impression that X-Force is better, since it is clearly better at classifying vulnerabilities with low access complexity, and only a little worse at classifying those with medium. However, as seen in Table 3, low access complexity is quite rare compared to medium, so NVD still wins out. Similarly, the fact that there are very few examples of vulnerabilities requiring adjacent network access and almost no vulnerabilities at all requiring multiple authentication mean that accuracy about these properties matters very little. Thus we can present a ranking of the databases in the form of Table 2. For example, the probability that NVD will make a correct guess on a randomly chosen metric of a randomly chosen vulnerability is 93%, while the same number is 79% for Cert-VN.

6.3 Joint prediction accuracy

Assuming that we have access to multiple databases, what is our calculated prediction accuracy? We are able to calculate this given the confusion matrices and Bayes theorem:

$$P(\mu|\chi_s) = \prod_s \frac{P(\chi_s|\mu)}{\sum_i P(\chi_s|\mu_i)} P(\mu) \quad (6)$$

As an example, before any observation, our prior probability of a randomly selected vulnerability having a `complete` confidentiality impact will be 21% (cf. Table 3). Learning that the NVD database scored the confidentiality impact to `complete`, our belief increases to 85%. Further confirming scorings from Cisco strengthens our belief to 98.7%, which is further raised to 99.4% with confirmation from X-Force, 99.99% with OSVDB and finally 99.999% when all five databases agree on complete confidentiality impact. However, if one database, say X-Force, disagrees with the other four databases and instead scores the impact to `partial`, our belief will decrease to 98.7%, and if OSVDB shares this dissenting view, the probability drops to 82%.

6.4 Overall distributions

One interesting kind of observation that can be made from the analysis is the overall distributions over the scales of the six CVSS dimensions. In other words, given the scorings of all the five databases, what do we think about the actual distributions of all vulnerability properties? These results are presented in Table 3.

7 Discussion

7.1 Reliability of results

One concern with the results presented in Section 6 is that the measurement method is not reliable, i.e. that it would

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2016.2644614, IEEE Transactions on Dependable and Secure Computing

TABLE 3
Overall characteristics of vulnerabilities, based on the five databases.

Metric	Values	Probability
Access vector	Local (L)	11%
	Adjacent Network (A)	0.3%
	Network (N)	88%
Access complexity	High (H)	58%
	Medium (M)	35%
	Low (L)	6%
Authentication	Multiple (M)	0.0001%
	Single (S)	6%
	None(N)	94%
Confidentiality	None(N)	33%
	Partial (P)	46%
	Complete (C)	21%
Integrity	None(N)	25%
	Partial (P)	54%
	Complete (C)	21%
Availability	None(N)	33%
	Partial (P)	42%
	Complete (C)	25%

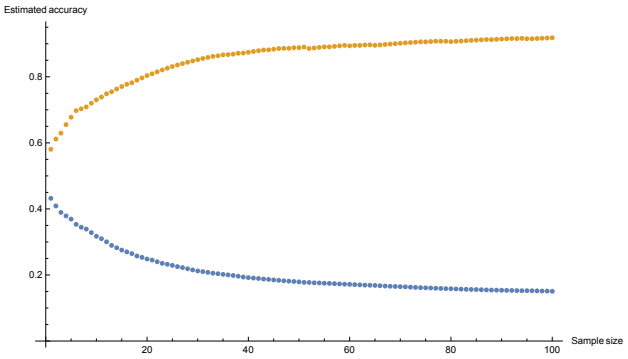

Fig. 8. Estimated accuracy of accurate and inaccurate scorers changing with the sample size in simulation. The upper curve plots the estimated credibility of the two identical scorers, while the lower plots the credibility of the single unaligned scorer.

not produce similar results if applied over and over again to similar data. In the following, a number of important reliability concerns are discussed.

In this study we use five different scoring instruments – the databases. If some of these are generally correct, but some are generally incorrect, will not the scores of the incorrect ones still affect our beliefs about the actual values, thus worsening reliability? It turns out that this is not the case. In a simulation, two scorers were set up to be completely aligned, scoring 90% complete impact in one of the CIA dimensions, while a third was unaligned and scoring only 10% complete impact. Fig. 8 displays how the estimated accuracy of the accurate and inaccurate scorers changes with the sample size in the aforementioned simulation. Initially, all scorers are equally credible, but the gradual accrual of evidence impacts scorer credibility (as well as beliefs about the actual CVSS score distributions). Thus, as the third scorer rarely matched the other two, its credibility eroded, thus shrinking the weight of its advise.

Another classic reliability concern is about the ability of the method to correctly measure the underlying distributions. To check this aspect of measurement reliability, we have performed simulation runs with fabricated data. We have found that the fabricated parameters (the Π and Ψ set by us) were faithfully recreated by the Gibbs sampler for all tested parameter choices.

Related to this is the question of whether the sample size is sufficient. It may seem unbalanced to have some databases provide only a few hundred samples, while others contribute with almost 40 000 samples. The concern would be that the massive datasets from NVD, X-Force and Cisco would overwhelm the moderate contributions of OSVDB and CERT. However, Fig. 8 indicates that database trustworthiness is established quite quickly, so a few hundred samples is indeed enough to establish the big picture.

A concrete example of a reliability concern related to sample size and imbalance is the fact that NVD never mistakes single-instance authentication for multiple instances, but often for no authentication. The reason for this is probably that there are no instances of multiple instance authentication where there is more than a single source. Thus, we have no case where NVD could be proven wrong. Strictly speaking, this is not a reliability shortcoming of the method as such, but rather of the underlying data material.

7.2 Validity of results

Another concern is that the results are not valid, i.e. that they do not measure what they are supposed to. In the following, a number of important validity concerns are discussed.

Perhaps the fundamental validity question is whether true CVSS values exist at all. Some dimensions indeed appear subjective. For instance, 'no authentication' is unambiguous, while 'medium access complexity' may be interpreted in different ways. Nevertheless, as the CVSS model was constructed to have unambiguous, correct values, and this model is widely used in practice, the statistical model of Section 4 assumes the existence of true values.

It might be argued that, in addition to looking at the assessments in the databases, we could find true CVSS values by analyzing vulnerabilities ourselves, demonstrating experimental validity. However, that would merely amount to one more observation by fallible observers. Indeed, being less familiar with vulnerability assessment than the experts, our observations would probably be *less* accurate than those of the databases. Thus the only feasible way to find the true CVSS scores is analysis of the databases.

This also relates to the discussion of joint bias above. As seen in the sample run shown in Fig. 8, the analysis is vulnerable to the case of several synchronised but incorrect databases. However, note that direct causal relations that *improve* rather than decrease the quality of databases are permissible. This actually improves our analysis. This is a key observation, as the database analysts are probably to look at each others verdicts to some extent, but also make their own assessments. If they agree, that would make them more convinced about their own verdict, but if they disagree, they simply set another score.

When would the model fail? When an analyst working at one of the vulnerability databases simply copies the verdicts of another database, because of incompetence or lack of time, *and* the copy is of a database that typically assigns the wrong score. Though possible, this is highly improbable. Thus in the majority of cases, our method should work fine.

As described in Section 5, we have good reason to believe in the independence of the five databases. Thus, while

influence on the assessment of one database from the others cannot be completely ruled out for every vulnerability, there is no reason to believe that there are any large and systematic dependencies between the databases.

A discussion of validity in a Bayesian setting is incomplete without addressing the priors. We use Dirichlet(3,1,1), Dirichlet(1,3,1) and Dirichlet(1,1,3) as priors for the states of each CVSS base score component. This captures the reasonable prior belief that a vulnerability with e.g. complete impact will be rated as such. The belief corresponds to the rational belief of having viewed three cases of that happening, but one case of partial being assigned to a complete-impact vulnerability and one case of none being assigned.

As discussed in Section 4.2, this prior distribution is weak, but not completely uninformative, as the prior distribution is the only thing making sure that e.g. 'complete' is measured as 'complete', rather than 'none', say. The Gibbs sampler does not care about variable names. (This is similar to two observers having different perceptions of red and green, for instance one perceiving red as the other perceives green, and vice versa. But as long as they *label* their perceptions so that the same objects are pointed out as red and green, this has no adverse practical consequences.) The inverted confusion matrix for access complexity in Fig. 5 (OSVDB is pretty good at consistently being wrong about access complexity) could have been avoided – but only at the cost of stronger priors. Stronger priors, however, is not desirable in a data-driven empirical study, where we wish to minimize the impact of our own preconceptions.

7.3 The changes between CVSS 2.0 and 3.0

Even though the foremost rationale for this study was to evaluate the credibility of the actual CVSS scores found in the databases, it is also possible to draw some tentative conclusions about the CVSS scoring system itself. In particular, it is interesting to contrast our findings with some of the criticisms raised about CVSS 2.0, not least those addressed in CVSS 3.0. For example, the Open Security Foundation (OSF) and Risk Based Security (RBS), the organizations behind OSVDB, raised a number of issues with CVSS 2.0 in the process of developing version 3.0.[14] Among the features of CVSS 2.0 criticized by OSF and RBS is the scale used for access complexity. The fact that this scale is problematic is also reflected in our results, where access complexity often shows considerable confusion. This was also acknowledged in the release notes of CVSS 3.0, where it was frankly stated that "Access Complexity is a very overloaded and subjective metric in CVSS v2".[15] Access Complexity was also among the categories that received the most suggestions for revision in the expert survey of Holm and Afridi [15].

7.4 Practical use of CVSS trust results

For academia it is important to know what data to trust. Many research studies (e.g. [1], [2], [3]) use CVSS scores from the databases investigated here – this study helps assessing the validity of such studies. As we have shown, the data can generally be trusted. Still, the details matter. Conclusions based on the confidentiality, integrity, and availability metrics are generally more trustworthy than those based on the other metrics, but CERT-VN CIA data is less accurate compared to the other databases. When looking at the six metric dimensions separately, data from different databases might be suitable in order to use the most trustworthy data. NVD, being most accurate in all dimensions, seems to be a clear choice to start with. If other databases are used as complement or second opinion one might want to avoid e.g. OSVDB for access complexity, and CERT-VN should not be the first choice when looking at CIA.

For industry, prioritizing vulnerabilities by severity can help save a lot of effort and money [7]. According to Symantec "many private-sector organizations are using CVSS internally to make informed vulnerability management decisions. [...] and remediate those vulnerabilities that pose the greatest risk to their systems."[16] Thus, having trustworthy information about vulnerabilities is key in order to make good prioritization decisions. In this respect, the needs of industry are similar to the academic needs discussed above.

The number of tool vendors focusing on security assessments is steadily increasing, and their information on vulnerabilities often comes from the databases studied. Thus, their reliability as vendors is dependent on the vulnerability information used. Some examples include the Product Security Incident Response Team at CISCO that uses CVSS as a major component in prioritizing their team workload[17], Tenable that employs CVSS scores together with the Nessus scanner results for many type of reports (e.g. risk matrices)[18], and Veracode that uses CVSS in e.g. their product VerAfied Security Mark[19].

8 Conclusions

In our Bayesian investigation of the CVSS it is clear that NVD in general performs very well. This is particularly true for the Confidentiality (C), Integrity (I), and Availability (A) dimensions. For the access complexity metric, a poorer result is seen when it comes to classifying vulnerabilities with low access complexity. For the authentication metric, we see great uncertainty when it comes to classifying vulnerabilities requiring multiple authentication.

X-Force also performs well in general. Compared to NVD, there are many similarities, but also differences. For instance, similar to NVD, there is great uncertainty when it comes to classifying vulnerabilities requiring multiple authentication, but unlike NVD, X-Force is strikingly poor at classifying vulnerabilities requiring single authentication, often classifying them as requiring none.

Yet other patterns are seen when looking at OSVDB. Again, the CIA impact dimensions are pretty good, but OSVDB still relatively often misclassifies vulnerabilities with partial impact as having complete. The classification of vulnerabilities exploitable from an adjacent location (access vector metric) is both poor and laden with uncertainty.

CERT-VN exhibits the weakest classifications of the CIA dimensions, but fares well when classifying vulnerabilities

14. http://www.riskbasedsecurity.com/reports/CVSS-ShortcomingsFaultsandFailures.pdf
15. https://www.first.org/cvss
16. www.symantec.com/connect/articles/introduction-common-vulnerability-scoring-system
17. www.cisco.com/c/en/us/about/security-center/cvss.html
18. www.tenable.com/sc-dashboards/cvss-base-risk-matrices
19. www.veracode.com/ratings

that need no authentication although the results for CERT-VN are associated with high uncertainties.

The results of Cisco are more certain. Cisco generally fares well in the CIA dimensions, though not as well as NVD. In the CIA dimensions, there is a consistent problem of misclassification of vulnerabilities with partial impact as having complete. The results for access complexity assessment are somewhat poorer than those of NVD and X-Force.

Marginalizing database accuracies allows single metric comparisons. The probability that a database will make a correct guess on a randomly chosen metric of a randomly chosen vulnerability is 93% for NVD, 91% for X-Force, 77% for OSVDB, 79% for CERT-VN, and 85% for Cisco.

Since academia, industry, and tool vendors all use CVSS data from the databases studied in this paper, our results are important. In academia, we need to know what data to trust in order for our studies to be valid. For industry and tool vendors, knowing that CVSS scores in general are trustworthy, and in particular what dimensions in which databases to trust the most, can help calibrate the decision support process when prioritizing risks and threats.

APPENDIX
CONFUSION MATRICES

In the following, numerical confusion matrices for each of the databases are given with respect to their classifications of each of the CVSS dimensions introduced in Table 1.

NVD

Access vector:

	N	A	L
N	0.99	0	0
A	0.21	0.71	0.08
L	0.05	0	0.95

Access complexity:

	L	M	H
L	0.54	0.29	0.17
M	0.02	0.88	0.1
H	0.01	0.08	0.91

Authentication:

	N	S	M
N	0.99	0.01	0
S	0.04	0.95	0.01
M	0.19	0.2	0.6

Confidentiality:

	C	P	N
C	0.92	0.07	0.01
P	0.05	0.9	0.04
N	0.03	0.08	0.9

Integrity:

	C	P	N
C	0.91	0.08	0.01
P	0.05	0.93	0.02
N	0.02	0.07	0.92

Availability:

	C	P	N
C	0.92	0.07	0.01
P	0.07	0.91	0.02
N	0.02	0.11	0.87

X-Force

Access vector:

	N	A	L
N	0.99	0	0.01
A	0.1	0.85	0.06
L	0.04	0	0.96

Access complexity:

	L	M	H
L	0.85	0.07	0.08
M	0.25	0.71	0.04
H	0.02	0.07	0.91

Authentication:

	N	S	M
N	0.99	0.01	0
S	0.66	0.34	0
M	0.25	0.2	0.55

Confidentiality:

	C	P	N
C	0.65	0.28	0.07
P	0.01	0.95	0.04
N	0	0.05	0.95

Integrity:

	C	P	N
C	0.66	0.32	0.02
P	0.01	0.97	0.03
N	0	0.05	0.95

Availability:

	C	P	N
C	0.66	0.29	0.05
P	0.02	0.95	0.03
N	0	0.03	0.96

OSVDB

Access vector:

	N	A	L
N	0.98	0	0.02
A	0.15	0.65	0.2
L	0.01	0.02	0.97

Access complexity:

	L	M	H
L	0.1	0.02	0.88
M	0.35	0.64	0.01
H	0.94	0.04	0.02

Authentication:

	N	S	M
N	0.96	0.03	0
S	0.29	0.69	0.01
M	0.21	0.19	0.6

Confidentiality:

	C	P	N
C	0.93	0.01	0.05
P	0.26	0.66	0.08
N	0.06	0.01	0.93

Integrity:

	C	P	N
C	0.95	0.03	0.03
P	0.19	0.77	0.04
N	0.08	0.02	0.9

Availability:

	C	P	N
C	0.93	0.02	0.05
P	0.29	0.7	0.01
N	0.05	0.04	0.92

CERT-VN

Access vector:

	N	A	L
N	0.9	0.08	0.02
A	0.26	0.63	0.11
L	0.32	0.17	0.51

Access complexity:

	L	M	H
L	0.79	0.1	0.11
M	0.04	0.77	0.19
H	0.02	0.23	0.75

Authentication:

	N	S	M
N	0.99	0	0
S	0.51	0.35	0.14
M	0.19	0.19	0.62

Confidentiality:

	C	P	N
C	0.89	0.08	0.03
P	0.31	0.66	0.03
N	0.13	0.33	0.54

Integrity:

	C	P	N
C	0.9	0.08	0.03
P	0.22	0.68	0.1
N	0.19	0.13	0.68

Availability:

	C	P	N
C	0.74	0.18	0.09
P	0.19	0.75	0.06
N	0.14	0.09	0.77

Cisco

Access vector:

	N	A	L
N	0.97	0.01	0.01
A	0.16	0.71	0.12
L	0.06	0.03	0.91

Access complexity:

	L	M	H
L	0.35	0.44	0.21
M	0.01	0.92	0.08
H	0.02	0.27	0.71

Authentication:

	N	S	M
N	0.84	0.15	0.01
S	0.06	0.92	0.01
M	0.21	0.2	0.59

Confidentiality:

	C	P	N
C	0.95	0.03	0.01
P	0.16	0.75	0.09
N	0.03	0.07	0.9

Integrity:

	C	P	N
C	0.96	0.03	0.01
P	0.13	0.78	0.09
N	0.03	0.05	0.92

Availability:

	C	P	N
C	0.96	0.03	0.01
P	0.27	0.65	0.08
N	0.05	0.05	0.9

ACKNOWLEDGMENTS

This project received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 607109. U. Franke was supported by the Swedish Civil Contingencies Agency, MSB, agreement no. 2015-6986.

REFERENCES

[1] S. Zhang, D. Caragea, and X. Ou, "An empirical study on using the national vulnerability database to predict software vulnerabilities," in *Database and Expert Systems Applications*. Springer, 2011, pp. 217–231.

[2] M. Abedin, S. Nessa, E. Al-Shaer, and L. Khan, "Vulnerability analysis for evaluating quality of protection of security policies," in *Proceedings of the 2nd ACM workshop on Quality of protection*. ACM, 2006, pp. 49–52.

[3] S. Huang, H. Tang, M. Zhang, and J. Tian, "Text clustering on national vulnerability database," in *Computer Engineering and Applications (ICCEA), 2010 Second International Conference on*, vol. 2. IEEE, 2010, pp. 295–299.

[4] Q. Liu and Y. Zhang, "VRSS: A new system for rating and scoring vulnerabilities," *Computer Communications*, vol. 34, no. 3, pp. 264–273, 2011.

[5] S. H. Houmb, V. N. Franqueira, and E. A. Engum, "Quantifying security risk level from CVSS estimates of frequency and impact," *Journal of Systems and Software*, vol. 83, no. 9, pp. 1622–1634, 2010.

[6] L. Gallon, "On the impact of environmental metrics on CVSS scores," in *IEEE SocialCom 2010*. IEEE, 2010, pp. 987–992.

[7] O. Rochford and L. Orans, "Threat and Vulnerability Management Primer for 2016," Gartner, Inc., Tech. Rep., Jan. 2016, G00293037.

[8] K. Scarfone and P. Mell, "An analysis of CVSS version 2 vulnerability scoring," in *Proc. 3rd International Symposium on Empirical Software Engineering and Measurement*. IEEE Computer Society, 2009, pp. 516–525.

[9] J. Pescatore, "Classifying and Prioritizing Software Vulnerabilities," Gartner, Inc., Tech. Rep., Sep. 2007, G00151699.

[10] L. Allodi and F. Massacci, "A preliminary analysis of vulnerability scores for attacks in wild: The EKITS and SYM datasets," in *Proceedings of the 2012 ACM Workshop on Building analysis datasets and gathering experience returns for security*. ACM, 2012, pp. 17–24.

[11] L. Allodi, W. Shim, and F. Massacci, "Quantitative assessment of risk reduction with cybercrime black market monitoring," in *Security and Privacy Workshops (SPW), 2013 IEEE*. IEEE, 2013, pp. 165–172.

[12] S. Zhang, X. Ou, and D. Caragea, "Predicting cyber risks through national vulnerability database," *Information Security Journal: A Global Perspective*, vol. 24, no. 4-6, pp. 194–206, 2015.

[13] H. Holm, M. Ekstedt, and D. Andersson, "Empirical analysis of system-level vulnerability metrics through actual attacks," *Dependable and Secure Computing, IEEE Transactions on*, vol. 9, no. 6, pp. 825–837, 2012.

[14] M. Bozorgi, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond heuristics: learning to classify vulnerabilities and predict exploits," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 105–114.

[15] H. Holm and K. K. Afridi, "An expert-based investigation of the common vulnerability scoring system," *Computers & Security*, vol. 53, pp. 18–30, 2015.

[16] A. Gelman, J. B. Carlin, D. B. Rubin, and H. S. Stern, *Bayesian data analysis*, 2nd ed. Chapman and Hall/CRC, 2004.

[17] J. Kruschke, *Doing Bayesian Data Analysis: A Tutorial Introduction with R*. Academic Press, 2010.

[18] M. K. Cowles, *Applied Bayesian statistics: with R and OpenBUGS examples*. Springer Science & Business Media, 2013, vol. 98.

[19] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, *Markov Chain Monte Carlo in Practice*. Springer Science & Business Media, 1996.

[20] M. Plummer, "JAGS version 4.0.0 user manual," 2015.

[21] P. Dalgaard, *Introductory statistics with R*. Springer Science & Business Media, 2008.

[22] M. Plummer, "rjags: Bayesian graphical models using MCMC," *R package version*, vol. 3, 2013.

[23] S. Wolfram, *The mathematica book*. Wolfram Media, Incorporated, 1996.

Pontus Johnson is a professor at the Royal Institute of Technology (KTH) in Stockholm, Sweden. He received his MSc from the Lund Institute of Technology in 1997 and his PhD from KTH in 2002. His research interests include enterprise and software architecture in general, and architecture analysis in particular.

Robert Lagerström is an associate professor at the Royal Institute of Technology (KTH) in Sweden and a visiting scholar at Harvard. He received his MSc degree in computer science and PhD degree in industrial information systems architecture in 2005 and 2010, respectively. His main research interests are IT management, enterprise architecture, system complexity, and cyber security.

Mathias Ekstedt is a professor at the Royal Institute of Technology (KTH) in Stockholm, Sweden. He received his MSc and PhD from KTH in 1999 and 2004 respectively. His research interests include software, systems, and enterprise architecture modeling and analyses, and information and cyber security.

Ulrik Franke is a senior researcher at the Swedish Institute of Computer Science (SICS). He received his MSc and PhD in 2007 and 2012, respectively, both from the Royal Institute of Technology (KTH) in Stockholm, Sweden. His research interests include enterprise architecture, decision-making, IT service availability and cyber security.