



Vulnerability forecasting—a conceptual model

H.S. Venter, J.H.P. Eloff*

Department of Computer Science, University of Pretoria, 0002 Pretoria, South Africa

Received 6 November 2003; revised 24 March 2004; accepted 9 June 2004

KEYWORDS

Vulnerability scanner (VS);
Vulnerability forecasting;
Harmonised vulnerability categories;
Vulnerability database;
Scan

Abstract Vulnerability scanners (VSs) are information security tools able to detect security weaknesses on hosts in a network. VSs secure hosts in a proactive manner. A proactive approach is considered to be better than reactive approaches followed by, for example, intrusion detection systems, because prevention is better than cure. There are many problems and disadvantages of currently available VSs, such as hampering system resources while conducting scans. This paper introduces a conceptual model for vulnerability forecasting. The model uses intelligent techniques to improve on the efficiency of currently available VSs. The model aims to do vulnerability forecasting specifically by predicting the number of known vulnerabilities that will occur in the near future by using intelligent techniques and vulnerability history data. The model is tested by means of a prototype and an evaluation of the model's results is also provided in the paper.

© 2004 Elsevier Ltd. All rights reserved.

Introduction

Currently there are various state-of-the-art information security technologies that can be used to secure computer systems and networks. A vulnerability scanner (VS) tool, also known as a vulnerability assessment tool, is an example of such an information security technology (Bace, 2000). Vulnerability constitutes any known weakness on

a system that could potentially be exploited by malicious software or hackers. VSs follow a *proactive* approach in finding and minimising network and system vulnerabilities because they scan for vulnerabilities and attempt to identify them before they can be exploited.

There are, however, many problems with state-of-the-art VSs. Since new vulnerabilities are identified on a daily basis, the vulnerability database of a VS will be outdated the moment new vulnerabilities are identified. It is therefore important to conduct VS scans on a daily basis, each time with the newly updated vulnerability information. However, it might not always be possible to conduct scans at regular intervals due to unforeseen

* Corresponding author. Tel.: +27 12 420 2361; fax: +27 12 362 5188.

E-mail addresses: hventer@cs.up.ac.za (H.S. Venter), eloff@cs.up.ac.za (J.H.P. Eloff).

circumstances, for example when critical maintenance is carried out on servers and the network.

While conducting a VS scan, the VS generally occupies a vast number of network and system resources, leading to the degradation of network and system performance. Furthermore, the kinds of vulnerabilities being scanned for differ extensively from one particular brand of VS to another. Most VSs provide lengthy rectification procedure reports on how to rectify the particular vulnerabilities that were found during a scan. Since the automation of these rectification procedures is far from achieved by current VSs, they provide inadequate and insufficient information, which would not currently aid vulnerability risk management.

This paper attempts to investigate these problems by proposing a model for vulnerability forecasting with a functional discussion of the model. In the remainder of the paper, the concept of vulnerability forecasting is defined, after which the model is introduced and functionally explained. The application of the model for vulnerability forecasting is then discussed after which the paper concludes with some future work.

Concept of vulnerability forecasting

The term “vulnerability forecasting” (VF) can be defined as an attempt to identify potential vulnerable areas on hosts across a network and the extent to which such areas on hosts across a network will be vulnerable over a specific period in the near future. The principal aim of VF is,

therefore, to predict trends or patterns in the number of known vulnerabilities that could occur.

A vulnerability forecast would, therefore, enable one to take proactive action in a bid to minimise the risks that such vulnerabilities may pose.

A conceptual model for VF

The conceptual VF model is shown in Fig. 1.

The conceptual VF model comprises three main components. The *current VS technology* component constitutes one or more state-of-the-art VS tools currently available in the software market, which are used for collecting the data needed for VF. The VS technology, therefore, is not revolutionary or new to the VF model, but rather existing technology that is utilised to collect and provide current vulnerability information to the VF model as a whole. The *vulnerability harmonisation* component serves as a coupler between the current VS technology component and the vulnerability forecasting component to transform the VS tool’s output into a harmonised form. The *vulnerability forecasting* component does the actual vulnerability forecast by predicting the number of vulnerabilities that will occur in the near future and by doing so would also provide which kinds of vulnerabilities should enjoy preference in rectifying them.

Each of the main components of the conceptual VF model contains subcomponents as depicted in Fig. 1. These are discussed in the sections that follow.

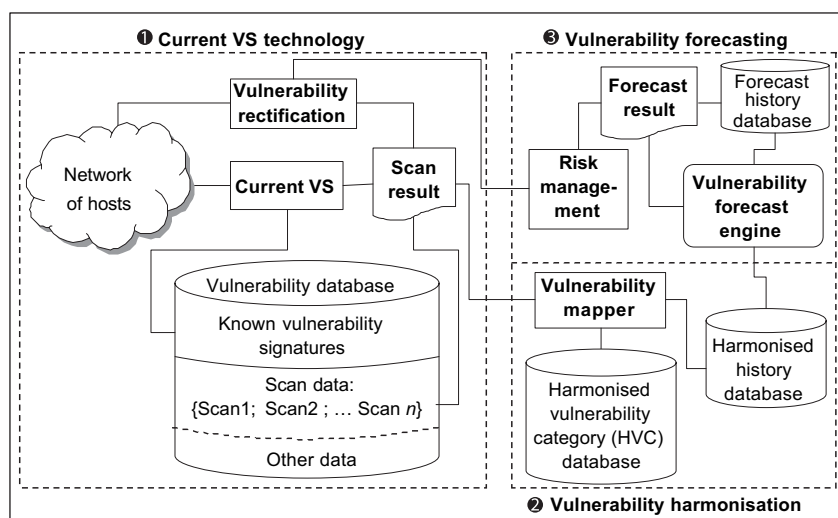


Figure 1 The conceptual model for vulnerability forecasting.

The current VS technology component

In this component of the VF model, one or more current VS tools attempt to detect vulnerabilities on a specified network of hosts. The network of hosts may also include hardware devices, for example routers, switches, hubs and network printers. Employing more than one VS may deliver more accurate results, because not all VSs scan for exactly the same vulnerabilities and, therefore, may improve the chances of identifying more vulnerabilities. For demonstration purposes, however, only one VS has been used in this model.

The VS uses a vulnerability database as a repository in which it stores the signatures of potentially all vulnerabilities known to date and should be kept up to date on a regular basis. A vulnerability is detected as soon as one of the vulnerability signatures in the vulnerability database is matched with an identical signature found on the network of hosts. The vulnerabilities that are detected by the VS are compiled in the form of a scan result report, which is also stored in the vulnerability database. The scan result undergoes vulnerability rectification, where skilled human resources can manually attempt to rectify these vulnerabilities. Scans are conducted at different time intervals and each time the particular scan result is added in the vulnerability database.

Some VSs might also support common vulnerabilities and exposures (CVE) ([The Mitre Corporation, 2003](#)) and give the appropriate CVE number for the specific vulnerability found. CVE is an organisation on the web that provides a list or dictionary of common names for publicly known information security vulnerabilities and exposures in an effort to standardise the naming of vulnerabilities, hoping to encourage their use in potentially all current and future VSs. Although this standard naming convention is a good initiative by CVE, the problem still persists of *which kinds* of vulnerabilities each VS can potentially detect. It is for this reason that the authors present another main component in the VF model—that of vulnerability harmonisation.

The vulnerability harmonisation component

The vulnerability harmonisation component of the conceptual VF model as shown in [Fig. 1](#) does not do the actual vulnerability forecast yet. Rather, it serves as a transitional process where the data it received from a scan is transformed in such a way that it is “harmonised” and, thus, prepared to

be “understood” by the vulnerability forecasting component of the conceptual VF model.

The output of the VS – the scan result of each scan – serves as input to the vulnerability mapper in the vulnerability harmonisation component. The vulnerability mapper maps the vulnerabilities found by the VS onto a harmonised set of vulnerability categories, known as the harmonised vulnerability categories (HVC). This set of vulnerability categories is stored in the HVC database. The HVCs were defined by the authors in previous work and a summary of this is shown in [Table 1 \(Venter and Eloff, 2003\)](#).














The result of the mapping process is stored in the harmonised history database, which is then used by the vulnerability forecasting component of the VF model. The harmonised history database contains the results of multiple scans.

Vulnerability forecasting

The vulnerability forecasting component does the actual vulnerability forecast. The output of the vulnerability harmonisation component, namely the harmonised history database, serves as input to the vulnerability forecast engine. The vulnerability forecast engine constitutes the heart of the conceptual VF model and attempts to predict trends or patterns, in terms of HVCs. For example, a typical forecast for a specific HVC entitled *network and system information gathering* may read as follows: “It is expected that a range of between x and y *network and system information gathering* vulnerabilities will be detected when the next scan is conducted.” A decision can then be made, as indicated by the risk management component in the VF model, as to whether or not the range of $[x, y]$ vulnerabilities for the said HVC poses a significant threat to an organisation.

This manner of VF can be facilitated by using a *Fuzzy Expected Interval (FEI)* ([Kandel, 1992b](#); [Schneider, 1988](#)), which forms a subset of fuzzy logic. An FEI for a specific HVC represents a narrow range of typical values. This range of typical values, such as $[x, y]$, best describes the possible number of currently known vulnerabilities for a specific HVC that can be expected to occur when a future scan is conducted. The reasons why the authors opted to employ an FEI include its ability to compensate for and deal with incomplete data. Incomplete data may be generated by real-world eventualities, for instance, when hosts are offline or when effecting software and hardware installations and removals, rendering a specific network to be in a different state than before.

Table 1 Summary of the harmonised vulnerability categories (HVCs)

	HVC	Brief description
1	 Password cracking and sniffing	Vulnerabilities with a root cause of having accounts with weak or no passwords
2	 Network and system information gathering	Vulnerabilities concerned with scanning a network to discover a map of available hosts and vulnerable services
3	 User enumeration and information gathering	Vulnerabilities concerned with retrieving information of user accounts on a specific system
4	 Backdoors, Trojans and remote controlling	Vulnerabilities concerned with having hidden access mechanisms installed on a system
5	 Unauthorised access to remote connections & services	Vulnerabilities concerned with the risk that an unauthorised person has the ability to connect to and misuse a system
6	 Privilege and user escalation	Vulnerabilities concerned with the risk that the access rights of an existing user account can be upgraded by an unauthorised user, granting more privileges to the user
7	 Spoofing or masquerading	Vulnerabilities concerned with the risk that an intruder can fake an IP address in a bid to act as another person
8	 Misconfigurations	Vulnerabilities concerned with the risk that applications have been incorrectly configured
9	 Denial-of-services (DoS) and buffer overflows	Vulnerabilities concerned with the risk of one or more intruders launching an attack designed to disrupt or deny legitimate users' or applications' ability to access resources
10	 Viruses and worms	Vulnerabilities concerned with malicious programs
11	 Hardware specific	Vulnerabilities concerned with having hardware peripherals that execute ROM-based or firmware-based programs
12	 Software specific and updates	Vulnerabilities concerned with the risk that specific applications contain specific, well-known bugs
13	 Security policy violations	Vulnerabilities concerned with the risk that an Internet security policy has been violated

An FEI is calculated for each HVC. Note that, in order to keep the vulnerability database up-to-date, the newest vulnerability signatures were downloaded and stored in the vulnerability database of the VS before each vulnerability scan was conducted. Consider that **10** scans have been conducted to obtain harmonised history data. Also consider only one specific HVC *K* over the **10** scans for now. Calculating the FEI for HVC *K*, and subsequently for all other HVCs, requires the following five steps:

- Step 1: Determine fuzzy vulnerability groups for a vulnerability forecast.
- Step 2: Defuzzify "fuzzy" vulnerabilities.
- Step 3: Define and calculate the vulnerability membership function.
- Step 4: Defuzzify "fuzzy" scans.
- Step 5: Calculate the fuzzy vulnerability group maximum over the minima to find finally the FEI.

These steps will be explained and demonstrated in detail in the sections that follow, based on example history vulnerability data as shown in Fig. 2.

Step 1: Determine fuzzy vulnerability groups for a vulnerability forecast

It is critical to know how the population is distributed across the harmonised history data. The population is distributed into certain fuzzy vulnerability groups. The term "population" is used to refer to the entire range of all possible values in the harmonised history data. Determining these fuzzy vulnerability groups is an intuitive exercise and can be effected, for example, by examining the history data, as shown in the graph in Fig. 2, for HVC *K* over the **10** scans. The specific number of fuzzy vulnerability groups to be determined is also defined intuitively – it depends on how closely the data are related. Each of the **10** scans must be allocated to a specific fuzzy vulnerability group. The idea is to group together the

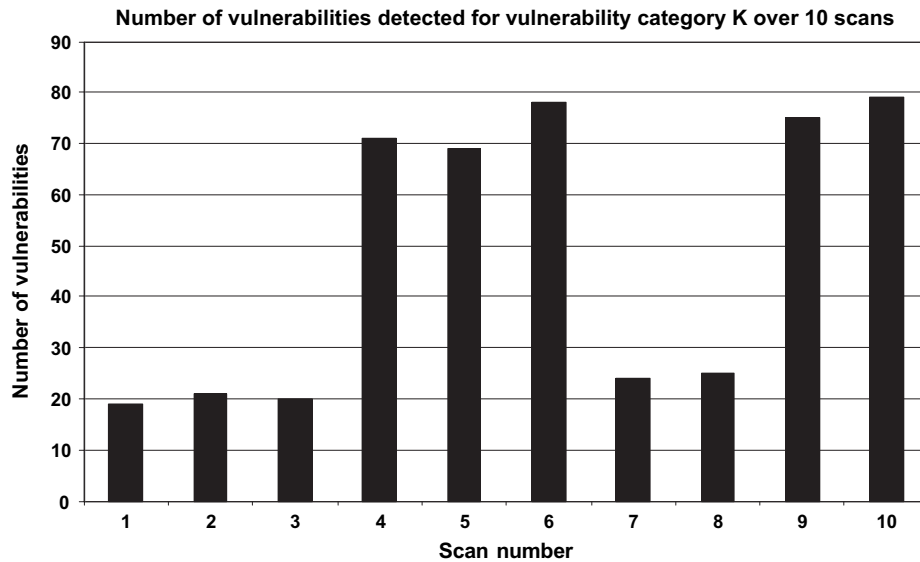


Figure 2 Graph with example vulnerability history data for 10 scans.

harmonised history data of those scans that closely relate to one another, for HVC *K*. In this way, the data in the 10 scans are summarised and lumped into fuzzy vulnerability groups (Kandel, 1992a). Table 2 shows examples of fuzzy vulnerability groups.

Note that the scans grouped together in fuzzy vulnerability group A fall *exactly* within the [20, 25] range. There is, in other words, nothing “fuzzy” about this range. Fuzzy vulnerability groups B and C, on the other hand, contain the adjectives *more or less* and *almost*, respectively, which render these ranges fuzzy, and not crisp. In the process of calculating an FEI, working with fuzzy values is precluded, as the calculations in question can only be effected through crisp logic. The adjectives *more or less* and *almost* need to be converted into ranges – they need to be defuzzified into crisp value ranges as explained in the following section.

Step 2: Defuzzify “fuzzy” vulnerabilities

The adjective “more or less 20” can be converted to an exact range containing a lower and an upper

Table 2 The fuzzy vulnerability groups formed for harmonised vulnerability category *K*

Fuzzy vulnerability group	Fuzzy vulnerability group description
A	In 3 scans, 20–25 vulnerabilities were found.
B	In 3–5 scans, more or less 20 vulnerabilities were found.
C	In 4–5 scans, almost 80 vulnerabilities were found.

bound by means of a vulnerability defuzzification mapping table as shown in Table 3. The formulae in Table 3 are also constructed intuitively by studying the harmonised history data gleaned from the 10 scans. The exact methods for doing this can be found in Schneider (1988).

Table 4 indicates the distribution of the scans as they were allocated to each fuzzy vulnerability group, as well as the defuzzified ranges as determined by the vulnerability defuzzification mapping table for the current example.

These defuzzified ranges now need to be converted again, using a vulnerability membership function. The latter conversion is necessary in order to obtain a normalised view of all the vulnerability ranges. This process is discussed in the section that follows.

Step 3: Define and calculate the vulnerability membership function

The vulnerability membership function simply expresses each of these defuzzified vulnerability ranges as a range between 0 and 1 and therefore expresses the grade of membership (Kandel and Byatt, 1978). Traditionally, the grade of membership 1 is assigned if the vulnerability range completely and fully belongs to the entire vulnerability population of 90 in this example, whilst 0 is

Table 3 Vulnerability defuzzification mapping table

Adjective	Lower bound	Upper bound
Almost	$x - 15\%$	$x - 3$
More or less	$x - 2$	$x + 10\%$

Table 4 Distribution of scans and defuzzified ranges

Fuzzy vulnerability group	Distribution of scans	Vulnerabilities found (defuzzified)
A	3 scans become [3, 3]	[20, 25]
B	3–5 scans become [3, 5]	[18, 22]
C	4–5 scans become [4, 5]	[68, 77]

assigned to a vulnerability range that does not belong to the entire vulnerability population at all. The greater the degree to which a vulnerability range belongs to the entire vulnerability population, the closer it is to a grade of membership to the entire vulnerability population.

The vulnerability membership function for this example would be constructed as shown in Fig. 3. As in the case of the fuzzy vulnerability groups and the vulnerability defuzzification mapping table, the vulnerability membership function is created intuitively. Assume that it is evident from the harmonised history data that not one of the 10 scans ever uncovered more than 90 vulnerabilities. This vulnerability membership function, therefore, indicates that 90 is the absolute maximum that will ever be reached for HVC *K* and hence 90 is referred to as the entire vulnerability population.

The results of this vulnerability membership function for each fuzzy vulnerability group will be referred to as χ_s , as shown in Table 5. For example, the χ range for the range [20, 25] is calculated first for the lower bound and then for the upper bound. This culminates in the χ range [0.222, 0.278] for fuzzy vulnerability group A, as shown in Table 5. The latter process is repeated for the rest of the fuzzy vulnerability groups.

The distribution of scans must now also be defuzzified to a fuzzy measure between 0 and 1, thus determining the degree to which a specific range of vulnerabilities belongs to the entire vulnerability population. This process is discussed in the following section.

$$\chi(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ x / 90 & \text{if } (0 < x < 90) \\ 1 & \text{if } x \geq 90 \end{cases}$$

Figure 3 Vulnerability membership function.

Table 5 Transforming the defuzzified values using the vulnerability membership function $\chi(x)$

Fuzzy vulnerability group	Distribution of scans	χ_s
A	[3, 3]	[0.222, 0.278]
B	[3, 5]	[0.2, 0.244]
C	[4, 5]	[0.756, 0.889]

Step 4: Defuzzify “fuzzy” scans

A fuzzy measure for the scan distribution of each fuzzy vulnerability group will be calculated and the results will be referred to as μ_s . Schneider (1988) uses two specific equations to calculate the μ_s for each fuzzy vulnerability group. These equations can be found in Appendix A. Having used these two equations to calculate each of the μ_s for each fuzzy vulnerability group, the results are as shown in Table 6.

The μ_s and χ_s in Table 6 can be explained, i.e. for fuzzy vulnerability group C, as follows. The defuzzified vulnerability range [0.756, 0.889] belongs [0.333, 0.455] to the scan population, and [0.333, 0.455], in a range between 0 and 1, means that this vulnerability range includes only [0.333, 0.455] times the entire scan population. The defuzzified vulnerability range [0.756, 0.889], thus, is included in very few of the 10 scans, owing to its membership of [0.333, 0.455] to the entire population of 10 scans.

The ultimate aim is to find a single range that would serve as the FEI for this specific HVC by calculating the median of all those ranges. This is done by calculating the maximum over the minima of all the ranges shown in Table 6, to be discussed in the section that follows.

Step 5: Calculate the fuzzy vulnerability group maximum over the minima to finally find the FEI

For the final step in this process, the $\max(\min(\mu_i, \chi_i))$ is calculated. The result for this example would yield [0.333, 0.455]. The latter value should be multiplied by 90 to express the range [0.333, 0.455] in “number of vulnerabilities”. This will

Table 6 Results for the μ_s and χ_s

Fuzzy vulnerability group	μ_s	χ_s
A	[1.0, 1.0]	[0.222, 0.278]
B	[0.7, 0.769]	[0.2, 0.244]
C	[0.333, 0.455]	[0.756, 0.889]

yield an FEI of [30, 41], which, in turn, serves to forecast that the next time a VS scan is conducted, it will uncover between 30 and 41 vulnerabilities for HVC K.

The vulnerability forecast was effected for HVC K only in the five steps above. It must also be effected for all the other HVCs, which will each yield an FEI. Such a forecast result is stored in the forecast history database along with previous forecasts made. The latest vulnerability forecast can be reviewed and proactive vulnerability rectification procedures can be carried out according to the specific vulnerability forecast’s guideline. The main purpose of the forecast history database, thus, is the same as that of the harmonised history database: to serve as a repository for storing history data about vulnerability forecasts that will be used as input for the next time a vulnerability forecast is made.

In order to test the VF model, a vulnerability forecasting prototype (Venter, 2003; Visser, 2003) was developed. The next section briefly discusses the application of the VF model by describing the prototype and its achievements, and also provides a critical discussion as to what extend the VF model solves the problems at hand.

Application of the VF model

The VF prototype

The specific test scenario for the VF prototype included a network of 59 hosts containing various operating systems. Vulnerability scans were

conducted daily for 15 consecutive days, after which 15 sets of harmonised vulnerability history data were available. A vulnerability forecast was made for each of the 13 HVCs. Another scan was conducted on day 16 and compared to the vulnerability forecast. In an effort to determine the durability of the forecast, another scan was done on day 45 and also compared to the forecast made on day 15. The forecast made for day 16 was almost spot-on when compared to the vulnerability scan for day 16. The same forecast compared to the actual vulnerability scan on day 45, however, was not as accurate, but still gives a fairly close indication of what to expect after 30 days of making a vulnerability forecast. These results are shown in the graph in Fig. 4.

The risk management component as indicated in Fig. 1 is not an automatic step, but rather an interactive step taken by human resources, which includes some decision-making on how vulnerabilities will be rectified. A forecast result, for example that shown in Fig. 4, will typically state that certain HVCs are high-risk categories, i.e. categories 2, 3, 5 and 8. It is then up to human resources to decide whether all vulnerabilities in the high-risk HVCs will be attended to for proactive rectification, or whether only certain vulnerabilities in these categories will be attended to for rectification. This step, therefore, would provide human resources with more efficient information, for example, on how to know where to start with rectification procedures, in terms of which harmonised vulnerability’s vulnerability forecast poses the biggest risk to the organisation according to the organisation’s needs.

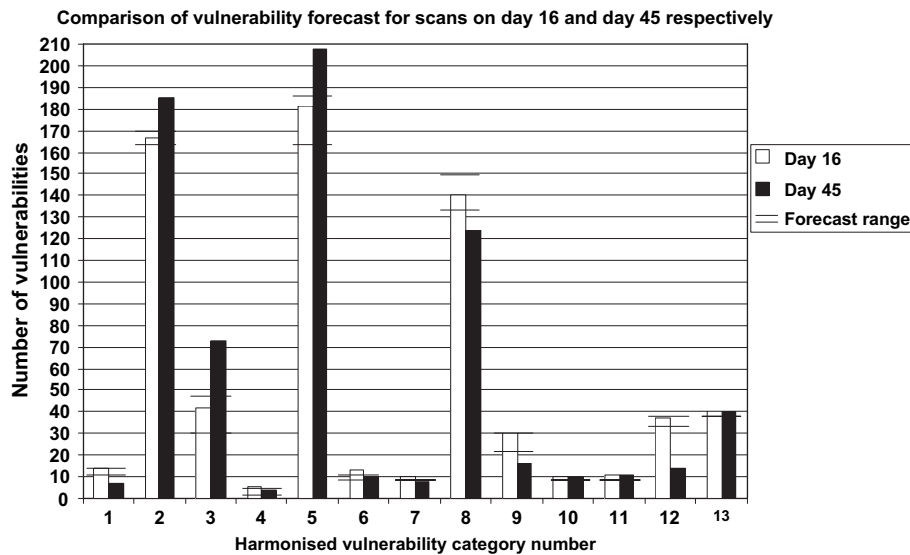


Figure 4 Scan results for day 16 and day 45 compared to the vulnerability forecast.

The improvements on VSs by the VF model

The principal aim of this paper is to make a contribution to proactive information security technologies, i.e. vulnerability scanning, in the Internet and network security domain. Since the VF model utilises existing VSs to conduct vulnerability scans, no improvement in vulnerability scanning tools themselves has been suggested. Instead, however, the VS process has been improved in that system resources are not hampered by frequent vulnerability scans, because the number of vulnerability scans conducted can be significantly decreased by making vulnerability forecasts. As a result of the latter, the administrative overhead of working through lengthy vulnerability scanner reports to rectify vulnerabilities is also significantly decreased. The main contribution of the VF model, therefore, enables one to forecast what the number of vulnerabilities will be for each HVC before the next vulnerability scan is conducted.

In addition to the above, the HVCs, as discussed in the vulnerability harmonisation component of the VF model, provide a standard method for grouping related vulnerabilities and, thus, enable one to know which subset of standardised vulnerabilities a specific vulnerability scanner can detect from a potentially exhaustive set of standardised vulnerabilities.

The main and final feature of the VF model is that it provides vulnerability forecasts in HVC format that would enable human resources to more effectively conduct vulnerability risk management and, thus, speeding up the vulnerability rectification process. There are, however, some limitations and problems of vulnerability forecasting that still need to be solved. These are discussed in the conclusion.

Conclusion

Having vulnerability forecasts means that vulnerability problem areas – in the form of HVCs – can be attended to before they can erupt. Furthermore, using HVCs along with vulnerability forecasting renders the process of doing vulnerability forecasting VS tool-independent.

As the case with almost any system, however, vulnerability forecasting has limitations. These limitations provide opportunities to extend and support the VF model by a number of future research projects. Further research is possible in a bid to find techniques on how to automate the procedure of mapping vulnerabilities of current VS

tools onto the HVCs, because this process is currently done manually. The process of configuring the vulnerability forecast engine according to the history scan data before a vulnerability forecast can be made is also currently done manually. Automating this process requires advanced techniques such as automatically setting up the fuzzy vulnerability groups from the history scan data, automatically creating the vulnerability defuzzification mapping table and automatically defining the vulnerability membership function.

Another possible future research project includes the integration of current VS technology and vulnerability forecasting in a single vulnerability-forecasting-enabled VS. In this way the administration of vulnerability forecasting will be decreased even further.

An interesting question to ask for future research is for *how long* will a vulnerability forecast remain valid? In addition, how much history data are necessary before one can really claim that it is enough to base a first vulnerability forecast on? These time-frame-based questions are questions that may be addressed in a research project of its own merit.

Vulnerability forecasting is built upon the specific information security technology called vulnerability scanning. It might be possible, as a future research project, to combine vulnerability scanning with other information security technologies such as intrusion detection systems and firewalls. In this way, hybrid vulnerability forecasting might be possible.

Finally, the VF model bases its vulnerability forecasts on *known* and *baseline* vulnerabilities, and not on new vulnerabilities that have not yet been identified. It would be fascinating, for a future research project, to have the VF model catering for both baseline and new vulnerabilities – being able to identify such vulnerabilities in an advanced manner and not only from using vulnerabilities that are currently known. A possible way to implement this is by looking for deviations from the currently known list of vulnerabilities. This way of vulnerability forecasting would likely be less accurate, but it would open up opportunity for interesting research, such as looking at the most active areas of vulnerability exploitation and exceptionally vulnerable kinds of operating systems.

Appendix A

Eqs. (1) and (2) below are used to calculate the lower bound and upper bound (μ_s) for each fuzzy vulnerability group, where n is the number of fuzzy

vulnerability groups identified and j is the current fuzzy vulnerability group in the calculation of LB_j (the μ for the *lower bound* of fuzzy vulnerability group j) and UB_j (the μ for the upper bound of fuzzy vulnerability group j)

$$LB_j = \frac{\sum_{i=j}^n \min [p_{i_1}, p_{i_2}]}{\sum_{i=j}^n \min [p_{i_1}, p_{i_2}] + \sum_{i=1}^{j-1} \max [p_{i_1}, p_{i_2}]} \quad (1)$$

$$UB_j = \frac{\sum_{i=j}^n \max [p_{i_1}, p_{i_2}]}{\sum_{i=j}^n \max [p_{i_1}, p_{i_2}] + \sum_{i=1}^{j-1} \min [p_{i_1}, p_{i_2}]} \quad (2)$$

References

- Bace RG. Intrusion detection. Defining intrusion detection. p. 3–4; Intrusion detection concepts. p. 37–43; Vulnerability analysis: a special case. p. 135–54. Macmillan Technical Publishing; 2000 [ISBN 1-57870-185-6].
- Kandel A, Byatt WJ. Fuzzy sets, fuzzy algebra and fuzzy statistics. Proc IEEE 1978;66(12):1619–31.
- Kandel A. Fuzzy sets, Fuzzy techniques in pattern recognition, John Wiley & Sons Inc.; 1992a [ISBN 0-471-09136-7; p. 23–43].
- Kandel A. General purpose fuzzy expert systems, Fuzzy expert systems, CRC Press Inc.; 1992b [ISBN 0-8493-4297-X; p. 23–41].
- The Mitre Corporation. CVE, the key to information sharing. Common vulnerabilities and exposures (CVE), 2003. <<http://www.cve.mitre.org/introduction.html>>.
- Schneider M. Properties of the fuzzy expected value and the fuzzy expected interval in fuzzy environment. Fuzzy sets and systems, vol. 28. Elsevier Science Publishers; 1988 [ISSN 0165-0114; p. 55–68].
- Venter HS, Eloff JHP. Harmonised vulnerability categories. South African Computer Journal 2003;29:24–31 [Computer Society of South Africa; ISSN 1015-7999].
- Venter HS. The VF prototype. Department of Computer Science, University of Pretoria, Pretoria, South Africa. <<http://www.cs.up.ac.za/cs/hventer/vf/vf.exe>>; hventer@cs.up.ac.za; 2003.
- Visser P. PO Box 746, Florida Hills, 1716, South Africa. The VF prototype. visser_pierre@hotmail.com; 2003.
- Dr. Hein S. Venter** graduated at the Rand Afrikaans University (RAU) in 1996, majoring in Computer Science and Informatics. He successfully obtained a BSc (Hons.) degree in Computer Science in 1997. In 1999 he received the MSc degree (Cum Laude) in Computer Science. He received his PhD degree in Computer Science at the RAU in 2003. Dr. Venter was appointed as a lecturer at the RAU from February 2000 until February 2003 in the Department of Computer Science at the RAU. Since February 2003 he has been a senior lecturer in the Department of Computer Science at the University of Pretoria. His research interests are in computer security, network security and Internet security. Dr. Venter has published in a number of accredited international subject journals. A number of acclaimed international and national, Computer and Information Security conferences, were attended where he presented his research papers. He is also a member of the organising committee of the Information Security for South Africa (ISSA) national conference and the South African Institute of Computer Scientists and Information Technologists (SAICSIT) national conference.
- Jan Eloff** received a PhD (Computer Science) from the Rand Afrikaans University, South Africa. He gained practical experience by working as management consultant specialising in the field of information security. Since October 2002 he is Head of Department and full professor in Computer Science at the Department of Computer Science, University of Pretoria. He is a representative from South Africa on Technical Committee 11 (Information Security) of the International Federation for Information Processing (IFIP). He has published extensively in a wide spectrum of accredited international subject journals. Many acclaimed international and national, Computer and Information Security conferences, were organised and conducted under his leadership. He is an evaluated researcher from The National Research Foundation (NRF), South Africa. He is a member of the Council for Natural Scientists of South Africa. He advises to industry on various information and computer security projects.

Available online at www.sciencedirect.com

SCIENCE @ DIRECT®