Categorizing vulnerabilities using data clustering techniques[1]

Yun (Lillian) Li　H.S. Venter　J.H.P. Eloff

Information and Computer Security Architectures (ICSA) Research Group

Department of Computer Science

University of Pretoria

Lynwood Drive
Pretoria
0002
South Africa

Tel: +27 12 420 3654
Email: {lli, hventer, eloff}@cs.up.ac.za

ABSTRACT

Vulnerability scanning is one of the proactive information security technologies in the Internet and network security domain. However, the current vulnerability scanner (VS) products differ extensively in the way that they can detect vulnerabilities, as well as in the number of vulnerabilities that they can detect. Often, VS products also declare their own vendor-specific vulnerability categories, which makes it difficult to study and compare them. Although Common Vulnerabilities and Exposures (CVE) provides a means to solve the disparate vulnerability names used in the different VS products; it does not standardize vulnerability categories. This paper presents a way to categorize the vulnerabilities in the CVE repository and proposes a solution for standardization of the vulnerability categories using a data-clustering algorithm.

KEY WORDS

Vulnerability, Vulnerability Scanners (VSs), Common Vulnerabilities and Exposures (CVE), Data clustering.

[1]This material is based upon work supported by the National Research Foundation under Grant number 2054024 as well as by Telkom and IST through THRIP. Any opinion, findings and conclusions or recommendations expressed in this material are those of the authors and therefore the NRF, Telkom and IST do not accept any liability thereto.

Categorizing vulnerabilities using data clustering techniques

1. INTRODUCTION

Nowadays, the Internet is a valuable part of our lives. It is used by millions of people everyday from all across the globe to perform online transactions, search for useful information and to communicate with other people. However, besides its power and value to us, its open and dynamic environment provides a perfect breeding ground for pernicious cyber-crimes and vicious security attacks. Five information security services are defined to implement security measures, namely, Authentication, Confidentiality, Integrity, Availability and Non-Repudiation [7]. Malicious attackers constantly look for and exploit weaknesses in computer systems in order to attack or break these security services. Those weaknesses in security systems that might be exploited to cause harm or loss are referred to as vulnerabilities [15]. This makes Internet security a challenging but interesting topic to research.

Many information security technologies have been developed by security experts, including: Cryptography [10], Firewalls [19] and Intrusion Detection Systems [2]. Each security technology implements one or more information security services mentioned above. Vulnerability Scanners (VSs), which are also referred to as vulnerability assessment technologies are proactive information security technologies and they attempt to search for known vulnerabilities before they can be exploited by intruders [22]. Despite the usefulness of VSs, there are some serious issues with current VSs. A major problem is that VSs are disparate in the ways that the vulnerabilities are named and organized in the vulnerability database of each different VS. For example, one VS might call a particular vulnerability a "Trojan horse" while another might call the same one a "virus". The number of vulnerabilities in the vulnerability databases of VSs also differs significantly. Furthermore, some VSs also define their own vulnerability categories. A vulnerability category refers to the grouping of specifically the same types of vulnerabilities. This complicates the problem even more. Some VSs define a small set of vulnerability categories while other VSs define many vulnerability categories. For example, SFProtect defines a hierarchy of seven vulnerability categories: User accounts, Audit policy, System logon, File system, Registry, Services, and Shares [18] where as TVAP defines DNS, CGI, Backdoors & Trojans, Databases, FTP, Brute Force Attacks, Firewalls, E-Commerce, General Services, All Network Appliances, Informational Services, Mail Services, SMTP & POP, SNMP, Web Servers (IIS, Apache, Zeus, Netscape, many more), Windows (9x, XP, 2000, NT), X Windows, Windows file sharing, Remote File Access, Denial of Service, NIS, Useless Services, and much more [20]. Lack of standards for vulnerability and vulnerability categories cause the disparity in the vulnerability database structures, which makes it very difficult to compare and assess VSs. Another problem with current VSs is that it creates huge administrative burdens. After each scan has been completed, a detailed report is generated to list all vulnerabilities found and the recommendations for the corrective actions. Such reports are often large and place huge burdens on administrators to rectify the vulnerabilities. For example, the Nessus Security Scanner [13] provides a report that gives detailed and organized information about all the vulnerabilities detected on the network based on the address of the hosts, ports and security issues regarding the port. It is still very hard for an administrator to go through it as it is too long and does not highlight the problem areas of the network for a quick response. This could have a negative impact on the efficiency and effectiveness of risk management, as vulnerabilities are often not rectified immediately.

In this paper, we are looking at the possibilities of using a data clustering technique to intelligently categorize vulnerabilities. We decided to use a data clustering technique because data clustering is a way in which we make clusters of objects that are somehow similar in characteristics in the sense that the intra-class similarity is maximized and inter-class similarity is minimized which is ideal for vulnerability categories. Another benefit of using data clustering in vulnerability categories is that minimal prior knowledge and assumptions are required and the number of categories does not have to be predetermined, as the data clustering technique will discover the hidden knowledge in the data set. This makes it simpler and different than the normal approaches where categories are identified first and vulnerabilities are then assigned to the most appropriate categories. We also suggest a solution for standardization of the vulnerability categories. This makes it possible to compare and assess VSs, and to increase the efficiency of risk management as more abstract reports can now be produced.

The rest of the paper is organized as follows: Section 2 presents background information on previous attempts to categorize vulnerabilities, Section 3 reviews the benefits standardizing vulnerability categories, and Section 4 discusses how to categorize CVE repository using a Self-Organizing Feature Map (SOM) followed by the conclusion in Section 5.

2. BACKGROUND

In the evolving world of computer and network security, continuous advances have been made in tool development and techniques for both compromising and protecting computer systems. The growth in the number and quality of security products indicates the perceived need for further means to protect computer systems from compromise [3].

Vulnerability scanning typically refers to the scanning of systems that are connected to the Internet but can also refer to system audits on internal networks that are not connected to the Internet in order to assess the threat of rogue software or malicious employees in an enterprise [23]. VSs are automated scanning programs that closely examine or scan computers and networks of computers to proactively detect known vulnerabilities [17]. While public servers are important for communication and data transfer over the Internet, they open the door to potential security breaches by threat agents, such as malicious hackers and intruders. VSs are proactive because they find vulnerabilities before an intruder has the opportunity to exploit them, thus they are preventative measures. VSs search for security flaws based on a database of known flaws, testing systems for the occurrence of these flaws and generating a report of the findings that an individual or an enterprise could use to tighten the network's security. The vulnerability database is updated periodically as new exploits are discovered.

Venter *et al.* has identified a set of 13 **Harmonized Vulnerability Categories,** which represent the entire range of vulnerabilities that are currently known [21]. The categories are: Password cracking and sniffing, Network and system information gathering, Backdoors, Trojans and remote controlling, Unauthorized access to remote connections and services, Privilege and user escalation, Spoofing or masquerading, Misconfigurations, Denial-of-services (DoS) and buffer overflows, Viruses and worms, Hardware specific, Software specific and updates, and Security policy violations.

The Harmonized Vulnerability Categories were identified by analyzing the security vulnerabilities found in current literature, current VSs and the Internet. This identifies the vulnerability categories based on the knowledge and expertise of the person or group that has to perform the categorization. Human expertise is required to manually assign the Harmonized Vulnerability categories to all the vulnerabilities. Although it is a once-off assignment, it is a tedious process.

Other than the Harmonized Vulnerability Categories, CISCO, which is one of the reputed Network Security companies, has also classified main vulnerabilities of systems into five categories. They are Design Errors, Protocol weaknesses, Software Vulnerabilities, Misconfiguration and Hostile code [4]. Many VSs also identified their proprietary vulnerability categories. For example, SAINT, which is a popular commercial VS product, has identified twelve vulnerability categories: Web, Mail, Ftp, Shell, Print, RPC, DNS, Database, Net, Windows, Passwords, and Miscellaneous [16]. Missouri Research & Education Network also identifies 25 vulnerability categories [12].

In this research we will discuss about an approach to categorizing vulnerabilities that eliminates the process of manually assigning vulnerability categories. The next section reviews the benefits of standardizing vulnerability categories and thus provides the foundation and the necessity for our research.

3. BENEFITS OF STANDARDIZING VULNERABILITY CATEGORIES

The main benefit of having a standard set of vulnerability categories is to enable the users to assess and compare VSs in order to determine their capabilities and pitfalls.

The benefits of standardizing vulnerability categories are: VS Assessment and Comparison, Interoperability and Abstract Reports. Each of the benefits is explained in the subsections below.

3.1 VS Assessment and Comparisons

Having a standard set of vulnerability categories is useful when conducting quantitative comparisons of various VSs. It allows us to investigate the number of vulnerability categories, as well as the number of vulnerabilities in each category that a VS can detect. This enables us to identify the strengths and weaknesses of a VS. It also facilitates the comparison between various VSs, which aids in making informed decisions when selecting the best suitable VS for an enterprise in terms of an enterprise's needs and priorities. These assessments and comparisons between various VSs allow one to incorporate multiple VSs in the enterprise in a meaningful way.

3.2 Interoperability

It could be possible to use an array of VSs can be used to protect the enterprise's network and assets where the weakness of one VS is the strength of another. The assessment of VSs allows for the best suitable tools to be selected to provide coverage without reliance on a single vendor for a "suite" solution. This in turn enhances the communication and security of organizations using VSs.

3.3 Abstract Reports

Abstract reports are produced which provide a high level view of the vulnerabilities detected. This allows the administrators to easily identify the problem categories or shortfalls in the network. Such reports allow more effective analysis for determining macro-level vulnerability patterns. The efficiency and effectiveness of risk management can also be improved since more abstract and comprehensive reports will be produced. Administrators can then rectify the detected vulnerabilities as soon as possible.

The next section presents our approach to categorize vulnerabilities that eliminates the process of manually assigning vulnerability categories.

4. CATEGORIZING COMMON VULNERABILITY AND EXPOSURES REPOSITORY USING SELF-ORGANIZING FEATURE MAP

One of the major problems with the current VSs as mentioned earlier is that VS vendors tend to name vulnerabilities differently. The CVE [11] is the internationally accepted naming standard for common vulnerabilities and exposures. It has become the de facto standard for information security vulnerabilities. Hence the following subsection discusses about CVE. SOM, the data clustering technique, is also discussed below which is the technique used in this research to categorize vulnerabilities in CVE.

4.1 Common Vulnerabilities and Exposures (CVE®)

The CVE was initiated in 1999 to solve this naming inconsistency. It is a list or dictionary that provides common names for publicly known information security vulnerabilities and exposures. The CVE repository is downloadable from the CVE web site [11] in HTML format, Text format, or Comma-separated format. Each entry in the CVE repository consists of three attributes: Name, Description and References, where Name can be considered as the primary key for the entry. Using a common name makes it easier to share data across separate databases and VSs that were not easily integrated. This makes CVE the key to sharing information security vulnerability information.

Although CVE is not quite a vulnerability database, almost all the major Vulnerability Databases (VDs) and VSs have a reference to it. The fact that it provides an internationally accepted naming standard for common vulnerabilities makes it a more suitable repository to perform clustering than any other vulnerability database.

According to CVE [11], there are currently 37 VDs, for example CERT/CC Vulnerability Notes Database, Cisco Secure Encyclopedia, and DeepSight Alert Service, as well as more than 50 VSs, for example Internet Scanner 6.5, Nessus Security Scanner and SAINT etc, worldwide that reference or intend to reference the CVE. Although only a few are actually CVE compatible at the moment, most of them declare CVE output and are CVE searchable. This means that a user can perform a search using a CVE name to find related information and the results presented will include the related CVE name(s). Thus by categorizing CVE entries, we are actually attempting to standardize the categorization of the vulnerabilities across different VDs and VSs.

The following section initially explains what a SOM is and then provides our approach of using this technique to perform categorization.

4.2 Self-Organizing Feature Map (SOM)

A Self-Organizing Feature Map (SOM) is a clustering tool that is useful for visualizing high-dimensional data in 2D space [6]. The benefits of using SOM will be discussed later in Section 4.4. It consists of a map of units and a set of input vectors known as the training set, as shown in Figure , where the circles symbolize the units. Each unit represents a weight vector that has the same dimension as the input vectors. The weight vectors can be initialized using various methods:

- Assign random values to each dimension of the weight vector. The initial random values are bounded by the range of the corresponding input parameter.
- Find the principal components of the set of input vectors and initialize the weight vectors to reflect these components.
- Initialize each weight vector to a randomly selected input vector.

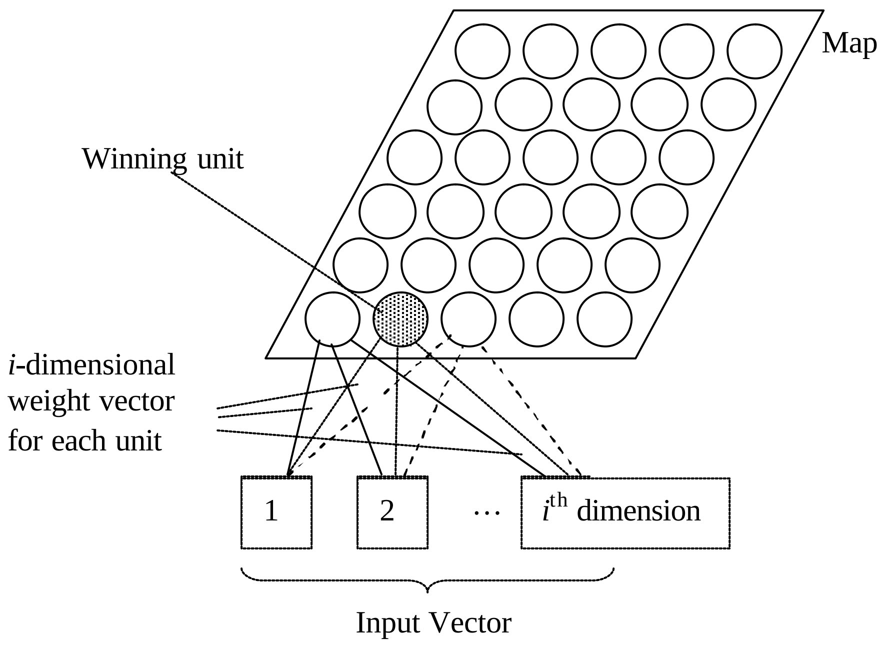

Figure 1. SOM Architecture

SOM training is based on a competitive learning strategy [1]. For each input vector ***v*** in the training set, the Euclidean distance [6] is calculated to each unit in the SOM. Each unit competes to match ***v***. The unit that is closest to ***v*** is known as the winning unit. The weights of the winning unit and those of its nearest neighbours are adjusted so as to reduce the Euclidean distance to the input vector. Adjusting the neighbors of the winning unit will allow for other vectors that are similar to ***v*** to be grouped together in the SOM. After training, similar patterns can be mapped to units that are close together in the SOM. This however, does not provide the cluster boundaries that specify the categories of the set of training vectors. To define the cluster boundaries, an additional step is required.

To determine cluster boundaries, the distance between each unit in the SOM can be calculated and stored in a matrix, known as the unified distance matrix (U-matrix) [6]. Large values within the U-matrix show the position of cluster boundaries. Another method to find cluster boundaries is the Ward clustering method [14]. This method initially assumes that each unit represents a cluster. Two clusters that are the closest to one another are merged at consecutive iterations. This is done until the optimal or specified number of clusters has been formed.

Before using the SOM tool, the dimension of the map needs to be determined. The dimension is measured by the number of units (for example, a 6x5 map has 6 rows of 5 units each). Usually the number of units in the SOM is less than the size of the data set. For example, the data set may contain 5000 training vectors but only 500 units are needed by SOM. If too many units are used, the SOM will overfit the input vectors; in addition, it may also cause undesirable small clusters. Overfitting implies that the SOM is memorizing the training set and will not be able to correctly classify new input vectors. Another side effect of too many units may cause the SOM tool to create proper clusters of similar training vectors, but many units have a zero or close to zero frequency. The frequency of a unit is the number of training vectors for which that unit is the winner. Alternatively, too few units will result in fewer clusters, which do not provide the optimal categories of the training vectors. This deli mar is resolved using a Growing SOM [9], in which the SOM architecture attempts to adapt to the training set. The Growing SOM first starts with a small number of units and then adapts by adding more units as needed, resulting in an architecture that is best suited for the training set.

The most important step before clustering using the SOM is the data preprocessing. The SOM tool uses a set of *i*-dimensional training vectors. Thus we cannot parse the natural language descriptions in the CVE directly to it, because the descriptions vary in length and are not of a numeric type.

The next subsection discusses the necessary data preprocessing to transform the data format of the CVE repository into a format that can be used and processed by the SOM.

4.3 Data Preprocessing

Data preprocessing techniques are used to improve the quality of the data so as to improve the accuracy and efficiency of the clustering process. There are various forms of data preprocessing [8]:

- Data cleaning fills in missing values, smoothes noisy data, identifies or removes outliers, and resolves inconsistencies.
- Data integration is the inclusion of multiple sources of data (databases, data cubes or files). Integrating many data sources may cause redundancies that will slow down the clustering process and this preprocessing step must take measures to ensure that these redundancies are removed.
- The data transformation step transforms or consolidates data into a form that is appropriate for clustering.
- Data reduction obtains a reduced representation of the data set that is much smaller in volume, yet produces almost the same analytical analysis.

After downloading the CVE repository from the CVE web site, it needs to be imported to a database for further processing. In this case, the comma-separated format of CVE was downloaded and imported to a Microsoft Access database.

Data integration was not necessary as the CVE repository is one data file. Data reduction was not performed since the CVE repository is not large.

For the data cleaning preprocessing step, common words and punctuation are removed from each entry in the description column of the CVE repository. The following rules are applied when deciding which word to add to the common words list:

- Remove all the preposition words such as by, without, when, and, that, etc.
- Remove all adjectives, adverbs, and verbs.
- Remove all the specific commands, application names and software version numbers.
- Remove all the words consisting of a single character.

To perform the clustering process, the description from the CVE entries must be transformed into a vector of numeric type. To achieve this, a set of words, *S*, is created from the description the entire CVE entries after the common words have been removed. For each entry in the CVE, a numeric vector is created which represents the occurrences of each word in *S*. For example, if *S* was the set of words {a, b, c, x, y, z} then an entry such as "a c c x z" will produce a numeric vector [1, 0, 2, 1, 0, 1]. That is one *a*, zero *b*, two *c*, one *x*, zero *y* and one *z*. Once this preprocessing step has completed, a set of numeric vectors represent the entries in the CVE. Each dimension in the vector represents a word that may cluster two or more entries in the CVE. This set of words thus provides a way to define a vector, which can be used across all the entries in the CVE repository.

The most important attribute of the CVE repository to us is the Description field, which contains the standard names given to the publicly known information security vulnerabilities in the form of natural language. For example, "*Buffer overflow in NFS mountd gives root access to remote attackers, mostly in Linux systems*" is the description given for the vulnerability Name CVE-1999-0002. Words such as "*in*" and "*to*" are common to many entries in the CVE and carry no specific meaning, thus these words can be removed to shorten the sentence. This will place more emphasis on important words such as "*Buffer*" and "*overflow*". By removing all the unimportant words, we have simplified the training process for the SOM and prevented clustering around common words.

The following section explains the reasons and benefits of using SOM.

4.4 Benefits of using SOM

We selected the SOM as a data clustering technique because it is an unsupervised learning neural network [5], which means that no target solution is needed beforehand. Thus we do not have to predetermine the number of categories. This implies that no prior knowledge is needed for the vulnerabilities and vulnerability categories. It will try to discover the pattern or knowledge hidden in the input data set. The results from SOM are easy to visualize and interpret [6]. This helps us when inspecting the categories that are formed. An example of a Cluster Map is shown in Figure 2. This map reveals four different clusters, a green, red, cyan and purple. The brightness of the color reveals the distance of the unit to the center of gravity. The center of gravity is the map unit, which most closely represents the average of all the units within a cluster. Brighter colors indicate a large distance, while darker colors a smaller distance to the center of gravity.

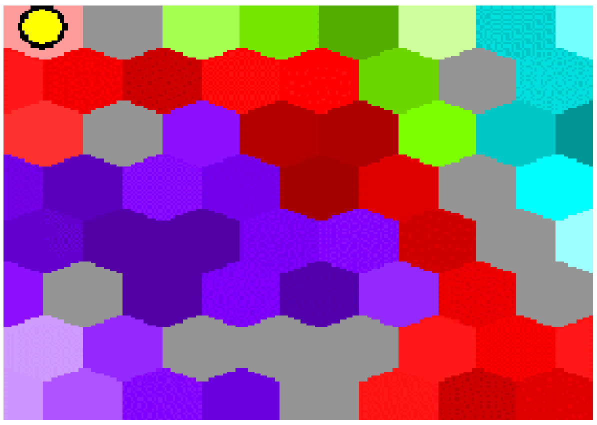

Figure 2. Cluster Map

SOM intelligently clusters vulnerabilities into different categories of similar nature. However, these clusters are not labelled and it needs to be done manually by careful inspection of the cluster's properties. The labeling procedure is done only once. When completed, a new entry can be mapped to the SOM and classified easily. This has no serious impact, as the clusters have to be inspected manually to discover what caused the SOM to form the cluster.

SOM offers a 2D visual plot of a multi-dimensional set of data. In addition to this image, each dimension of the weight vectors in the SOM can be visualized. This means that a new image (a component map) can be produced that shows the distribution of a dimension of the weight vectors. An example of a Component Map is shown in Figure 3. The Component Maps reveal the value variation of components/attributes over the map. It is the combination of all these components that determine the formation of clusters seen in the Cluster Maps. In Figure 3 the blue color indicates small values, while the red indicates larger values for a specific component.

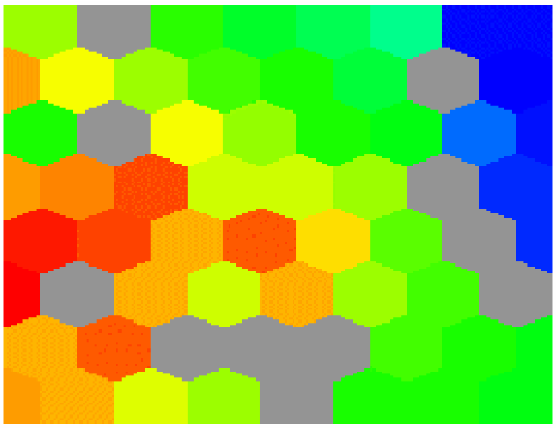

Figure 3. Component Map

For example, if we wanted to see the distribution of the dimension that represents the word "access" then a component plane can show where the word "access" is on the SOM and in which category it belongs to. A Combination of words may also be used, for example, we could find the distribution of the words "access" and "apache" on the same map. It is also possible to mark a region on the SOM and project the region onto a component map. Using the SOM and component plane to investigate the patterns within the data is known as exploratory data analysis.

5. CONCLUSION

In this paper, we looked at the possibilities of categorizing vulnerabilities in the CVE using SOM. This approach appears to effectively eliminate the tedious and error-prone human process of classifying vulnerabilities. It has also proposed a solution to standardize vulnerability categories as the categories formed are based on CVE, which is considered to be the standard for vulnerabilities. The advantage of using SOM in our approach is the easy visualization and interpretation of the clusters. At the time of writing this paper, we are still busy training the SOM and therefore no concrete results are yet available.

Once the training is complete, the future plan is to develop an automated process to assess VSs and produce more high-level reports from current VS scan reports. Although the concept of VSs and intrusion detection is different in the sense that VSs is proactive and intrusion detection is reactive in detecting intrusions, VSs also use signatures to identify vulnerabilities which are similar to signature- based intrusion detection systems. Thus, this approach can be extended into Intrusion Detection Systems to standardize the categories of intrusions as another future research goal.

6. REFERENCES

[1] Angeline, P. J. and Pollack, J. B., *Competitive environments evolve better solutions for complex tasks*. In Forrest, S., editor, Proceedings of the Fifth International Conference on Genetic Algorithms, pp. 264-270, San Mateo, CA. Morgan Kaufmann, 1993.

[2] Bace R.G., *Intrusion Detection*, pp. 37-43, Macmillan Technical Publishing, 2000, ISBN 1-57870-185-6.

[3] Baker D. W., Christey S. M., Hill W. H., Mann D. E., *The Development of a Common Enumeration of Vulnerabilities and Exposures*, Second International Workshop on Recent Advances in Intrusion Detection, 1999.

[4] Kujawski P, *Why Networks Must Be Secured*, Cisco Systems, Inc., 2003.

[5] DE SA V. R., *Unsupervised Classification Learning from Cross-Modal Environmental Structure*. PhD thesis, Department of Computer Science, University of Rochester. also available as TR 536, 1994.

[6] Engelbrecht A. P., *Computational Intelligence, An Introduction*, pp. 61-71, John Wiley & Sons, Inc., 2002, ISBN 0-470-84870-7.

[7] Gollmann G., *Computer Security*, pp. 5-9, John Wiley & Sons, Inc., 1999, ISBN0- 471-97844-2.

[8] Han J., Kamber M., *Data Mining: Concepts and Techniques*, pp. 105-130, Morgan Kauffmann Publishers, 2001, ISBN 1-55860-489-8.

[9] Kirk J. S., Chang D. J., and Zurada J. M., *A Self-Organizing Map with Dynamic, Architecture for Efficient Color Quantization*. Proceedings of the International Joint Conference on Neural Networks, Washington, D.C., 2001.

[10] Menezes A.J., Van Oorschot P.C., Vanstone S.A., *Handbook of Applied Cryptography*, CRC Press, 1996, ISBN 0-8493-8523-7.

[11] MITRE Inc., http://www.cve.mitre.org/, 27/04/2004

[12] MOREnet, http://www.more.net/services/rva/categories.html 02/05/2004

[13] Nessus, http://www.nessus.org/ 29/04/2004

[14] Pedersen, T., and Bruce, R., *Distinguishing word senses in untagged text*. Proceedings of the Second Conference on Empirical Methods in Natural Language Processing, pp. 197-207, 1997.

[15] Pfleeger C. P., *Security in Computing*, Prentice Hall, 2003, ISBN 0-13-035548-8.

[16] SAINT Corporation, http://www.saintcorporation.com/ 02/05/2004

[17] Schneier B., *Secrets and Lies, Digital Security in a Networked World*, pp. 194-197, John Wiley & Sons, Inc., 2000, ISBN 0-471-25311-1.

[18] SFProtect, http://www.winnetmag.com/Article/ArticleID/8401/8401.html 28/04/2004

[19] Tiwana A., *Web Security*, pp. 112-135, Digital Press, 2000, ISBN 1-55558-210-9.

[20] Trellisnet, http://www.trellisnet.com/Security/tvap/tvap.asp 28/04/2004

[21] VENTER, H.S.; ELOFF, J.H.P., *Harmonising Vulnerability Categories,* South African Computer Journal; pp. 24-31; No. 29; Computer Society Of South Africa South Africa, 2002, ISSN 1015-7999.

[22] Venter H. S., *A Model For Vulnerability Forecasting*, PhD thesis, Faculty of Natural Sciences, Rand Afrikaans University, 2003.

[23] WEBOPEDIA, http://www.webopedia.com/TERM/V/vulnerability_scanning.html 25/04/2004