ORGANIZE VULNERABILITIES BY DATA CLUSTERING TECHNIQUES

Bhanudas S.Panchabhai[1]* AND Dr.Ashok N.Patil[2]

[1]* Department of computer science, R.C.Patel Arts, Science and Commerce College, Shirpur
Dist.Dhule (Maharashtra)
Email-id :bharat.panchbhai@gmail.com
[2] Vasantrao Naik Arts, Science College, Shahada, Dist.Nandurbar (Maharashtra)
Email-id:Patelan71700@rediffmail.com

Abstract

Vulnerability scanning is one of the optimistic information security technologies in the Internet and network security area. However, the current vulnerability scanner (VS) products vary 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 category, which makes it difficult to study and compare them. Although Common Vulnerabilities and Exposures (CVE) provides a means to solve the different vulnerability names used in the different VS products; it does not standardize vulnerability categories. This paper presents a way to classify 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), Self organization map(SOM),Vulnerability Databases(VDs).

ORGANIZE VULNERABILITIES BY DATA CLUSTERING TECHNIQUES

1. Introduction

The first world has moved to computer empowered information based economies and is preparing to move past A watershed point from a state where networks, technology and the Internet facilitate public services, commerce and consumer needs, to one where public services, commerce and consumer needs are entirely dependent upon networks, technology and the Internet. The level of 'interconnectedness' the world is experiencecing is unprecedented. Rates of growth for data usage and proliferation of information processing systems and computerized devices are constantly on the increase. By the end of 2012 the number internet users was 4.2 billion people. It is used by millions of people every day from all across the globe to perform online transactions, search for useful information and communicate with other people. Though even with its influence and worth to us, its unlock and energetic atmosphere provides a perfect reproduction for malicious cyber crimes and unkind security attacks. That can seriously change or harm the confidentiality of data, the integrity of data, and the availability of systems **[1]**. Those weaknesses in security systems that might be exploited to cause harm or loss are referred to as vulnerabilities **[2]**. This makes Internet security a challenging and interesting topic to research.

In the growing world of computer and network security. Constant progress has been made in tool development and techniques for both compromising and protecting computer systems. Security experts have developed many information security technologies for **example**: Firewalls [TIWA 00] **[3]**.Intrusion Detection systems [BACE 00] **[4]**.Vulnerability scanners [HARN 02] and cryptography [MCCL 02] **[5]** .Each security technology implements one or more of the information security services mentioned above.

Vulnerability scanners which are also referred to as vulnerability assessment technologies are positive information security technologies and they attempt to search for known vulnerabilities before these can be exploited by intruders [VENT 03] **[6]**. Despite the usefulness of Vulnerability scanners, there are some serious issues with current Vulnerability scanners. A major problem is that Vulnerability Scanners are different in the ways that the vulnerabilities are named and organized in the vulnerability database of each different vulnerability scanners.

For **example**: vulnerability scanner **A** is able to detect only a certain type of vulnerability, whereas vulnerability scanner **B** can detect different types of vulnerabilities. Due to such disparity, vulnerability scanner assessment and comparison cannot be carried out in a uniform way. Since there exist no real measurement for vulnerability product assessment. Another problem with current Vulnerability scanners 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 **[7]** 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 potential of using a data clustering technique to sharply 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 past 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 Vulnerability scanners, and to increase the efficiency of risk management as more abstract reports can now be produced.

The rest of the research paper is organized as follows: **Section 2** present background information on previous attempts to categorize vulnerabilities **Section 3** reviews the benefits standardizing vulnerability categories, and **Section 4** discuss how to categorize CVE repository using a Self Organizing Feature Map (SOM).

2. Background

In the growing world of computer and network security, development of tools and techniques are continuously improved to both compromise and protect computer systems. The increase in the number and quality of security products indicates the need for further ways to protect computer systems from compromise [BAKE 99] **[8]**.

Many vulnerability scanners contain their own vulnerabilities databases, and others use vulnerability databases from well-accredited external sources. A vulnerability database is a comprehensive database of vulnerabilities that contains information on vulnerabilities .It may also contain additional information such as links to patch information or corrective actions for the vulnerabilities. Vulnerabilities information is critical for the protection of information systems. For **example**: Enterprise need to know whether components of their existing or planned computing environment are vulnerable to security failure and software developers need early warnings when their products have security flaws. A standardized and comprehensive vulnerability list is valuable to security experts for them to build products and services accordingly, and

system administrators want information on relevant security flaws and their resolutions. The quality of the vulnerability scanners is directly dependent on the quality of its vulnerability database. Furthermore, by exchanging, interpreting and correlating information about vulnerabilities among various scanners, collaboration can be achieved. It is obviously very hard to achieve this. One of the major obstacles to achieving collaboration is the lack of a common enumeration of computer vulnerabilities. Many vulnerabilities scanners adopt their own naming scheme for their vulnerabilities. As a result, it causes disparity among the scanners, which makes it very difficult to compare and assess them. As referred to earlier, one vulnerability scanner might call a particular vulnerability a "Trojan horse", while another might call the same vulnerability a "virus".

Venter *et al.* has identified a set of 13 Harmonized Vulnerability Categories**,** which represent the entire range of vulnerabilities that are currently known **[6]**. 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.

Other than the Harmonized Vulnerability Categories, CISCO, this 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 **[9]**. Many Vulnerability scanners also identified their Proprietary vulnerability Categories. For example, SAINT, which is a popular commercial vulnerability product, identified twelve vulnerability categories: Web, Mail, Ftp, Shell, Print, RPC, DNS, Database, Net, Windows, Passwords, and miscellaneous **[10].** Missouri Research & Education Network also identifies 25 vulnerability categories **[11]**.

In this research we will discuss about an approach to categorizing vulnerabilities that eliminates the process of manually assigning vulnerability categories. The next section presents 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 advantage of having a standard set of vulnerability category is to enable the users to evaluate and compare Vulnerability scanners in order to determine their capability and pitfall. The benefits of standardizing vulnerability categories are explained in the following subsections below.

3.1 VS evaluation and comparison

Having a standard set of vulnerability categories is useful when conducting quantitative

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

3.2 Interoperability

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

3.3 Conceptual information

Abstract reports are produced which provide a high level view of the vulnerabilities detected. This allows the administrators to easily identify the problem categories 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 vulnerability scanners as mentioned earlier is that Vulnerability scanners vendors tend to name vulnerabilities differently. The CVE **[12]** is the internationally accepted naming standard for common vulnerabilities and exposures. It has become the effectively standard for information security vulnerabilities. Hence the following subsection discusses about CVE. Self organization Map (SOM), the data clustering technique, is also discussed below which the technique is 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 **[12]** in HTML format, Text format, for 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 Vulnerability scanners 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 Vulnerability scanners 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 [13], there are currently 37 VDs, for **example**:Security Focus,CERT/CC Vulnerability Notes Database, Cisco Secure Encyclopedia, and Deep Sight Alert Service, as well as more than 40 Vulnerability scanners, 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 Vulnerability scanners.

The current research focuses on providing a standard set of vulnerability categories through the use of a data clustering tool as self organization map.

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)

The self-organizing feature map (SOM) is a method of unsupervised learning, based on a grid of artificial neuron units whose weights are adapted to match input vectors in a training set. It was first described by the Finnish professor Teuvo Kohonen and is thus sometimes referred to as a Kohonen map [KOHO 95] **[13]**.

The SOM algorithm is fed with i-dimensional feature vectors, where i represent the number of dimensions. In most applications, however, the number of dimensions will be high .Output maps can also be generated in different dimensions (1-dimensional, 2-dimensional, etc.), but most popular are two and three dimensional maps because SOMs are mainly used to reduce the problem space into a two or three dimensional space that is more understandable to humans.

SOM is a clustering tool that is useful for visualizing high-dimensional data in two-dimensional space [ENGE 02]. The benefits of using SOM will be discussed in the section that follows .It consists of a map of artificial neuron units and a set of i-dimensional input vector known as the training set (See in Figure 1) , where the circles in the map symbolize the neuron units. In this figure, it is a 7×9 (7 rows & 9 columns) map that contains 63 neurons units. Each unit is a weight vector that has the same dimension as the input vectors.

The following is a summary of the **SOM algorithm** [KOHO 95] **[13]**.

- Randomize/Initialize the weight vectors of the map's units.
- Read an input vector.
- Traverse each unit I the map:
    - Use the Euclidean distance formula to calculate the distance between the input vector and the unit.
    - Track the node that produces the smallest distance (this node will be called the Best Match Unit).
- Updates the units in the neighborhood of BMU by adjusting their weights so that they can be closer to the input vector.
- Repeat the process until the termination condition is met.

The termination condition can be the maximum number of times or epochs to repeat the above mentioned process, or if the calculated error is less than a certain value. This error is the map quantization error, which is calculated after each training iteration. The error describes the accuracy or "how good" the map is trained, with smaller values indicating a better result. After Training, similar patterns can be mapped to units that are close together in the SOM.

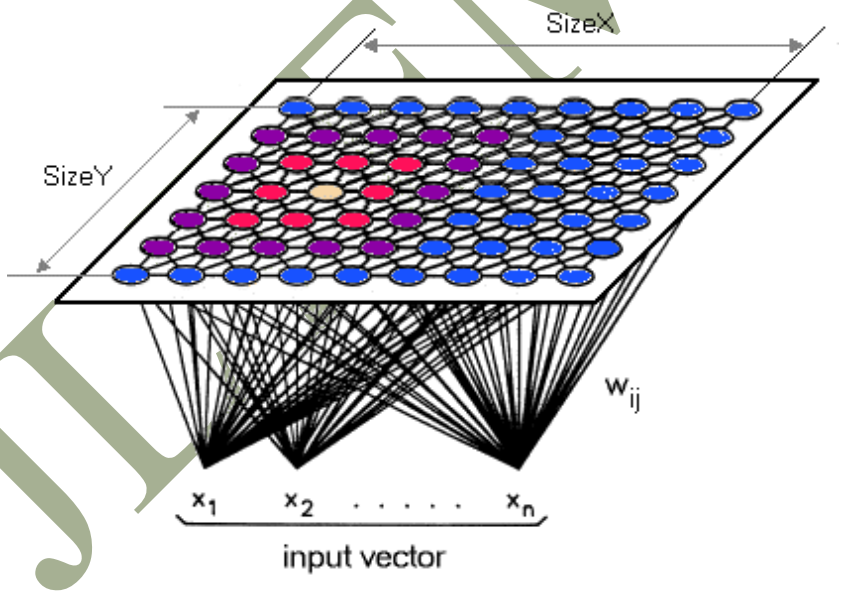

Figure 1 SOM Architecture

Using Figure1 the SOM algorithm can be easily explained in terms of a set of artificial neuron units – each having its own physical location on the output map-that take part in a winner-take-all process . A unit with its weight vector closest to the vectors of inputs (according to the Euclidean distance) is declared the winner and its weights are adjusted making them closer to the input vector. Each node has a set of neighbours.The weights of the defined neighbors of the winning unit are also adjusted to a lesser degree. The further the neighbor is from the winning unit, the less the weight is adjusted. This

process is then repeated for each input vector for a number of training iterations .Different inputs produce different winners. The SOM associates units with groups or patterns in the input data set. For labeled patterns, the labels can be attached to the associated units in the trained network SOM training is based on a competitive learning strategy **[14]**. For each input vector ***v*** in the training set, the Euclidean distance **[13]** 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 neighbors 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 vector s. 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) **[13]**. Large values within the U-matrix show the position of cluster boundaries. Another method to find cluster boundaries is the Ward clustering method **[15]**. 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 over fit the input vectors; in addition, it may also cause Undesirable small clusters. Over fitting 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 **[16]**, 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 **[17]**:

- Data clean-up fills in missing values, smoothes noisy data, identifies or removes outliers, and resolves inconsistencies.
- Data incorporation 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 mount 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

Below is a list of the reasons for selecting the SOM as the clustering tool for this research. The Reasons are discussed in depth in the paragraphs that follow.

- SOM is unsupervised learning; therefore no prior knowledge of the problem is required.
- SOM is highly scalable.
- SOM is able to deal with noise and outliers.
- SOM is insensitive to the order of the input records.
- SOM can handle high dimensionality.
- SOM aids in interpretability and usability.
- SOM aids in Visualization.

The SOM is selected because it is an unsupervised learning neural network, which means that no target solution is needed beforehand. As explained in earlier no prior knowledge needed for the vulnerabilities and vulnerability categories implies that the end result is not biased. The SOM will try to discover the pattern or knowledge hidden in the input data set. The ability to handle high dimensional space is very important to this research as the researcher is trying to cluster natural language descriptions. Natural language has approximately a million words, thus the dimensions of the data set will be huge. The input pattern can be in any random order, and SOM is able to deal with noise, outliers and missing values in the data set .The SOM is also scalable.

The results from SOM are easy to visualize and interpret **[13].** 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.

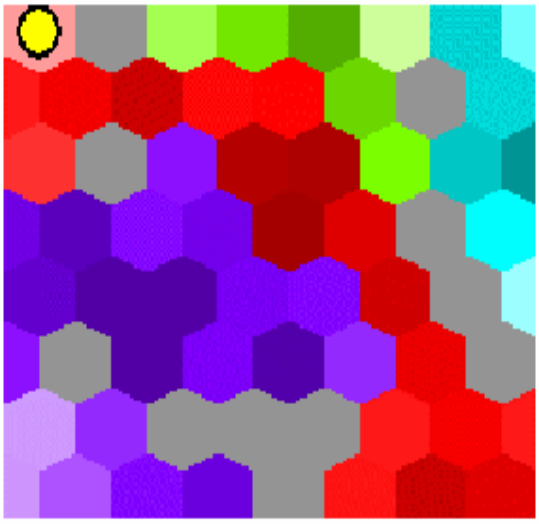

Figure 2. Cluster Map

SOM intelligently clusters vulnerabilities into different categories of similar nature. However, these clusters are not labeled 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.

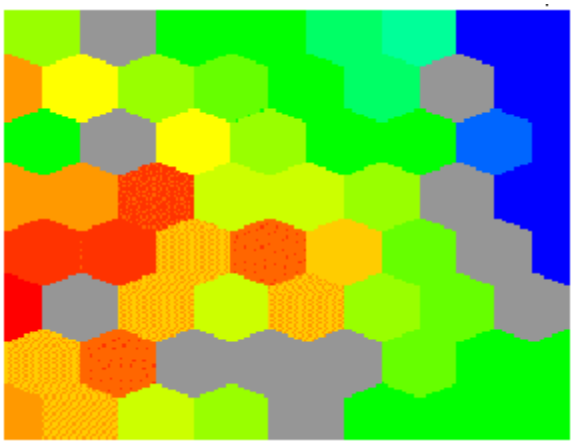

Figure 3. Component Map

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 research paper discuss the strengths and weaknesses of the vulnerability tool or database under assessment. Here we looked at the possibilities of categorizing vulnerabilities in the CVE using SOM.This approach appears to effectively eliminate the monotonous and error-prone human process of classifying vulnerabilities. It has also planned 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 understanding of the clusters.

6. Acknowledgement

The authors are grateful to Prof.B. V. Pawar, Head, department of computer science, North Maharashtra University, Jalgaon and Dr.Ruchira Bhargava, Coordinator, JJTU, Rajasthan .The author (B.S.Panchabhai) is thankful to Chairman Shri.Amarishbhai Patel and Principal Dr.D.R.Patil, R.C.Patel ASC College, Shirpur for their Valuable guidance.

7. References

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

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

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

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

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

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

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

[8] 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.

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

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

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

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

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

[14] 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.

[15] 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.

[16] 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

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