Network Security Assessment Using Data Flow Modeling

ABSTRACT

Network security is now an important factor for every organization, and became a mission-critical concern. Today, increasing number of threats demands continuous follow-up for each network component to ensure a reasonable security level. Our approach aims to simplify the task of risk assessment on network administrators by redrawing the real world network as a graph based on Data Flow Model (DFM). The nodes and links of the graph represent the components of the network. The analysis of the network will be based on administrators answers of checklists associated with each component.

Keywords

Network Security, Data Flow Model (DFM), Security Assessment.

1. INTRODUCTION

Network security is a challenging problem. A number of approaches were proposed to address this problem. A graph –based approach was proposed to analyze the vulnerability of networks. The Approach used a conjunction of the physical network topology with a set of attacks [4]. Another Approach based on chaining of exploits with the assumption of monotonicity [5]. On the other hand, some approaches consider the analysis of network security configuration [6,7]. The correct configuration of the network components is a major challenge especially when considering connected networks.

Our work introduces the use of data flow. The network will be laid as a graph based on the flow of data. The idea behind this approach is to apply checklists against the DFM to generate an over all security assessment based on score calculations. This approach helps administrators to assess the security of their network based on the application layer. The approach consists of three main steps. The first step is that the administrator draws the DFM of his network. The second step, is answering checklists associated with each component within the DFM. The checklists will vary for each component based on the component type, and should be updated regularly. The last step, when all checklists are answered, is generating the security evaluation report that contains detailed information about DFM. The evaluation will be calculated based on the administrators' answers of the checklists [1].

The rest of this paper will describe this approach in more details. Section 2 will contain an overview of our approach. It will describe the three main steps that are mentioned before in detail and how they are related to each other.

The next section will explain how the ranking is calculated and contains the formulas used to generate the ranking report. Section 4 contains the details about the tool that we designed based on the proposed approach.

We conclude with some suggestions for future work and the work can be extended.

2. Approach Overview

In our approach, the administrator is required to draw the network topology based on the flow of data. The graph shows all components and connections between them. By this step, the administrator can visualize the network as a topology, so the network administrator can discover if there are any illogical arrangements among the graph. Section 2.1 will describe the first step of the approach, which is drawing the DFM.

Checklists' answering is the second step in our approach, and it is the base of our ranking formulas (ranking methods will be discussed in more detail later in section 3). Therefore administrators must answer all the questions of the checklist as precise as possible to ensure an accurate score. At the end of this section we will describe how to build and maintain the checklists, followed by description of the security report generation.

2.1 Step 1: Drawing the DFM

Any network consists of different paths that represent how the data is moved around the network. On each path we find several network components connected to each other by a connection medium.

In our case, a network component is any thing that can be found on a network either a device or software. Examples for network components can be Web Server, Mail Server, Router, Switch, Firewall, Client Computers…etc.

A connection medium is any type of communication carrier that allows data to be transferred from one component to another. Such

connections include Twisted Pair cables, Coaxial Cables, Wireless Connections, Fiber Optic…etc.

In order to enable the administrator to draw, he must be given the proper notation for the graph to be clear and understandable. There are two notations to be used, one for representing a component and another for representing a connection. This notation must be able to distinguish between different types of components and different types of connections.

While drawing the network, the connections are being laid. Each connection can be either single direction or bidirectional based on the flow of data. The direction of the connection represents how the data is being transferred from one component to another.

Drawing a network needs a good knowledge about the network being drawn (i.e. what components are there in the network and what type of connections between them).

The information of each component supported by the software is best being stored in an XML file. A sample of the XML file is found in Figure 1 [2].

After completing the graph, the DFM is complete and ready to be an input for Step 2.

```
<components>
<category name="Web Servers">
<icon file="iis.jpg" checklist="IIS" Description="Internet Information Services" />
<icon file="apache.jpg" checklist="Apache" Description ="Apache" />
</category>

<connections name="Connections">
<type checklist="Wireless" Descriptoin="Wireless" file="wireless.ico" color="black" isdashed="1" />
<type checklist="Twisted Pair" Description="Twisted Pair" file="twisted.ico" color="black" isdashed="0" />
</connections>
</components>
```

Figure 1: Part of the XML components file.

2.2 Step 2: Answering the Checklists

By checklists, we mean a list of questions associated with all possible answers. Each component has its own checklist that investigates the security state of that component. The result of answering the checklist based on that component configuration is the current security state of the component.

Each checklist is gathered from different sources to form that checklist. Some sources include the vendor's website and documentation of the component that can include some checklists or important notes on how to configure that component to ensure a high security level. Other sources include the books that describe the component. Moreover, the security alert bulletins that contain useful information such as newly discovered bugs and vulnerabilities.

A checklist for a specific component contains a list of questions. For each question, there is a list of possible answers. Each answer is associated with a ranking value that represents the importance of this answer in configuring this component. The last thing that is recommended to be associated with the question is a description of that question and a recommendation of the best answer. This description can be helpful for the administrators to be aware of the best configuration that need to be applied.

After collecting the checklists from different sources, we saved them in XML format. In addition, since there's no predefined tag set in our approach, there cannot be any preconceived semantics. All of the semantics of an XML document will be defined by the applications that process them and apply our approach, which improves the functionality of the applications that uses the checklist, providing more flexible and adaptable information identification. A suggested formation of that XML file is found in Figure 2 [2].

```
<checklist name="Internet Information Services Generic">
<questions>
<question qid="1">
<questionstring>Are the log files analyzed regularly?</questionstring>
<questionanswers>
<answer answerid="1" rank="7">Yes</answer>
<answer answerid="2" rank="3">No</answer>
</questionanswers>
<why>Log files contain the systems activity. So it is important to monitor them regularly for unauthorized use of the system.</why>
</question>

<question qid="2">
<questionstring>Are the services on the computer running with the least-privileged accounts?</questionstring>
<questionanswers>
```

```
<answer answerid="1" rank="7">Yes</answer>
<answer answerid="2" rank="0">No</answer>
</questionanswers>
<why>Giving a service an administrative privilege may allow the attacker who gains access to that computer to act as the administrator of the system.</why>
</question>
</questions>
</checklist>
```

Figure 2: Part of the Checklists XML file

2.3 Step 3: Ranking Calculations and Report

After all the checklists are answered, the system begins analyzing the network. It divides it into paths where a path begins with a starting point and ends with an ending point. A starting point is components that dose not receive any incoming connections from other components and has at least one outgoing connection to another component. An ending point is components that dose not have any outgoing connection to any other component and at least one incoming connection. Figure 3 shows the definition of the starting and ending points where "A" is a starting point and "B" and "C" are ending points. Therefore, from the figure we conclude that there are two paths, "A→B" and "A→C".

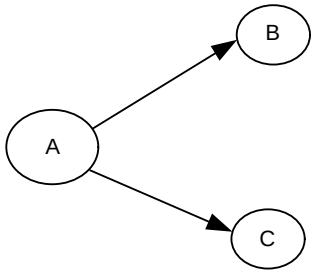

Figure 3: Starting and Ending Points Description

After determining all the paths that exist on the drawing, we start calculating the ranking for each path. The ranking for each path is determined by first getting all the ranking for each component within that path, and then depending on the location of the lowest component, the ranking is calculated. After calculating the ranking for each path, the overall ranking for the whole graph is calculated. After calculating the overall ranking, the security ranking report is generated. The detailed ranking formulas are described in the next section.

The security ranking report contains three types of information [2]. The first type is the overall ranking for the whole drawing. When viewing this type this ranking of the drawing along with the ranking of the highest and lowest path and component.

The second type of information is related to paths. After selecting which path to view, it is drawn in the drawing area and the ranking for that path is displayed along with the ranking of the highest and lowest ranking within that path. In addition, if a firewall is needed and missing in the path, a warning is displayed.

The third and last type is the ranking for each component. After selecting a component, its ranking is displayed and a list of checklist review list that contains questions that are answered incorrectly is displayed so the administrator can improve the security of his network.

3. Ranking Formulas

Ranking decision is one of important steps in our tool, because the decision should reflect the results of administrator interaction with the tool, and this result should be very accurate in describing the current security status of the administrator network from our perspective. For this reason, we try to provide ranking methodologies that apply logical and mathematical sense in giving the final decision in a percentage form.

First, we define five levels of ranking, each associated with a value. At first, we thought about giving unacceptable answers for checklist negative values; but we faced problems of getting negative percentage for the ranking decision, which is unaccepted mathematically. Because of that, we decided to use positive values starting form zero, which represent a very bad answer, ending with ten, which represent a very good answer [2].

Table 1 shows the levels with corresponding values.

Table 1: Ranking Levels

Value	Description
10	Very Secure Choice
7	Preferable
5	Neutral (no effect on security)
2	Not Preferable
0	Can Compromise Network Security

Yet to avoid unreasonable results caused by giving bad answers positive values we define four

ranking degrees based on the resultant percentage. The worst degree includes all percentages less than sixty and defined as “Very bad” and the best degree includes all percentages more than ninety and defined as “Excellent” [2].

After the administrator completes all the questions related to a particular component. A ranking of that component is displayed indicating the percentage of the security level of that component. In addition, after completing all the questions of all the components within the graph, the security of the different paths among the graph is calculated, and the overall security level is calculated too. The reason that we are doing this (giving three ranking percentages to the administrator) is that he can know where the weak points of his network are. If we only give him the overall ranking and it was very low, he may not know where the problem is. However, giving each component a separate ranking, he can distinguish between the secure components and non-secure components. This decomposition means that we need to have three separate formulas to calculate the ranking. These formulas are [2]:

3.1 Component Ranking

The formula for calculating the ranking for a component can be expressed as the summation of the rank of the answers chosen by the administrator, divided by the summation of the best ranking for each answer set, and then multiplied by 100 to get the percentage [2].

For example, if there were two questions as follows:

Are the log files analyzed regularly?

With possible answers of:

Yes with ranking of 7.

No with ranking of 3.

Are the services on the computer running with the least-privileged accounts?

With possible answers of:

Yes with ranking of 7.

No with ranking of 0.

Suppose the administrator answered the first question with “Yes” and the second question with “No”, the ranking for the component having this checklist is:

$$\frac{7+0}{7+7} * 100 = 50\%$$

3.2 Path Ranking

As mentioned before, a path is a set of components that starts with a starting point and ends with an ending point (section 2.3).

The formula for calculating the ranking over a path is complicated, so we will divide it into three parts, which are [2]:

3.2.1 Component within the Path Ranking

The components ranking is the summation of the ranking for each component on the path, and for the component with the worst ranking will be tripled if this component is the starting point and doubled otherwise. Calculating the worst component more than once is important to represent the affection of this component on the overall security of the path, because if we take the average, this will omit the affection of the worst component. We triple the ranking of the worst component if it is a starting point because in addition to the previous reason the insecure starting point can compromise the entire path [2].

3.2.2 Connections within the Path Ranking

The connections ranking is the summation of the ranking of each connection on the path, and the connection with the worst ranking will be doubled to ensure affection on the overall connections ranking. This summation is divided by the number of connections on the path considering the duplication of the worst connection. This will result on a percentage value we consider it as the ranking of the connections component [2].

3.2.3 Overall path ranking

Now, if the ranking percentage of the connections component is less than the ranking percentage of the worst component, then double the ranking of the connection component, otherwise, just add it to the summation. Finally, calculate the summation of the components ranking and the connections component ranking and divide it by the number of components on the path plus the connections component considering the number of duplications, this will result on a percentage value representing the path ranking [2].

To summarize the path ranking, we will describe it mathematically as follows where

Table 2 describes the variables used within the formulas [2]:

Table 2: Definition of variables used in overall path ranking formula

Term	Description
$Comp_{sum}$	The sum of the ranking of the components.
n	The number of the components in the path.
$Ranking_{comp\ i}$	The ranking of component i that was calculated according to the answers of the checklist.
Worst comp	The component with the smallest ranking in the path multiplied with a factor.
$Ranking_{worst\ component}$	The component with the smallest ranking in the path.
Factor	Equals to 2 only if the worst component is the first component in the path, otherwise it equals to 1.
$Connection_{sum}$	The sum of the ranking of all connections plus the ranking of the worst connection.
$Ranking_{connection\ i}$	The ranking of connection i that was calculated according to the answers of the checklist.
$Ranking_{worst\ connection}$	The connection with the smallest ranking in the path.
c	Number of connections.
$Ranking_{connections\ component}$	The average of the connection sum.
Total Path Ranking	The final sum of the ranking of the path.

$$let \quad comp_{sum} = \sum_{i=1}^{n} ranking_{comp_i}$$

$$let \quad worstcomp = ranking_{worst\ component} \times factor$$

were $factor = 2$, only if worst component is a starting point,

or $factor = 1$ *otherwise* .

$$let \quad connection_{sum} = \sum_{i=1}^{c} ranking_{connection_i} + ranking_{worestconnection}$$

then

$$ranking_{connections\ component} = \frac{connection_{sum}}{c+1}$$

Now we have two cases :

$$if\ ranking_{worstcomp} > ranking_{connections\ component}$$

$$ranking_{connections\ component} = ranking_{connections\ component} \times 2$$

and Total path ranking =

$$\frac{comp_{sum} + worstcomp + ranking_{connection\ s\ component}}{n + factor + 2}$$

else

Total path ranking =

$$\frac{comp_{sum} + worstcomp + ranking_{connection\ s\ component}}{n + factor + 1}$$

3.3 The overall graph ranking

For the overall graph ranking, we calculate the summation of the all paths ranking then, multiplying the worst path ranking by the number of paths minus one. Then we divide this summation to get the average result as a percentage value representing the overall ranking of the graph. This can be expressed mathematically as follows where represents the description of variables used within the formulas [2]:

Table 3: Definition of variables used in overall graph ranking formula

Term	Description
$Path_{sum}$	The sum of all paths in the graph.
np	The number of paths in the graph.
$Ranking_{path\ i}$	The ranking of path i that was calculated from the path ranking formula.
Worst path	The path with the smallest ranking multiplied by the number of all other paths.
$Ranking_{worst\ path}$	The path with the smallest ranking.
Graph ranking	The final ranking of the graph.

$$let\ path_{sum} = \sum_{i=1}^{np} ranking_{path_i}$$

$$let\ worstpath = ranking_{worstpath} \times (np - 1)$$

$$then,$$

$$Graphranking = \frac{path_{sum} + worstpath}{(2 \times np) - 1}$$

4. Design Overview

The system consists of several components shown in Figure 4. The update component is the process responsible of updating the checklists and the components. The process will be described in section 4.3.

The checklist bank component is the place where the checklists are stored (as an XML file) and accessed. The description of the checklists was explained in section 2.2 [2].

The components bank contains the components that are used within the software. Each component is associated an image to express that component, the name of the component, its type (Web Server, Mail Server…etc) and the checklist that is associated with it. The connections components are also stored in the bank where it has every thing as a normal component but instead of the image, it has the color of the connection and whether the drawn line is dashed represented a wireless connection or not. The bank is represented in an XML file shown before in Figure 1 [2].

The drawing system is the process responsible of displaying the network to be drawn. The process is also responsible of maintaining the coordinates of each component and what incoming and outgoing connections that are related to the component. This will be described in detail in section 4.1 [2].

Checklist answering process is responsible of retrieving the checklist associated with the component on the drawing the administrator wants to answer, and then display the questions for the administrator to answer. After answer all the questions, the answers are stored and associated with that component. This process is described in section 2.2 [2].

Drawing and Answers File is where the drawing and the answers of the checklists are stored permanently. In addition, since the file contains sensitive information (the weaknesses of the network), the file is encrypted. The encryption method will be described in section 4.2 [2].

The last part of the overall design is the Security Ranking Report. The report is a detailed description of the ranking for each component, path and the overall graph. It also contains a list of questions that are answered incorrectly so the administrator can correct them and improve the security level of his network [2].

The overall design is shown in Figure 4 [2].

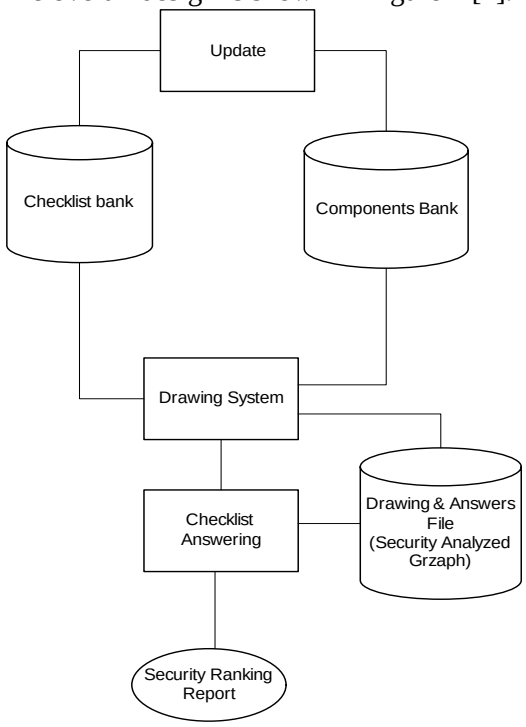

Figure 4: Overall Design

4.1 Drawing System

Drawing the DFD of the network is a main function in our tool. That is why we gave it a high priority in our work. When creating a new graph, the administrator can find a white area that accepts drawing on it. To add a new computer component on the graph, the administrator needs to drag the component he needs from the list of components and drop it on the drawing area. When a component is dropped, the component is drawn on the drawing area. In addition, a label below the image icon and the image icon are set to the same information found in the components XML file (see section 2.1) [2].

To draw connections in the graph, the administrator can drag a connection from the list of connections then click the starting and ending points of the connection. When drawing a connection, all the

proper properties are set such as the line color and line style so the line can take its shape. These properties are described in the XML file shown in Figure 1 [2].

The drawing system also supports notifying the administrator about components that have updated checklists or components that do not have any answered checklist. The notification can be done by changing the background of the image of the icon so when viewing the graph; the administrator can know what components do not have a complete answered checklist [2].

Figure 5 shows a graph drawn in the drawing area. On the left of the drawing area is the list components supported by the software. On the left of the drawing area are some important functions, which are answering all incomplete checklists and generating the security report such as the one shown in Figure 6 that represents the report for the graph drawn in Figure 5 [2].

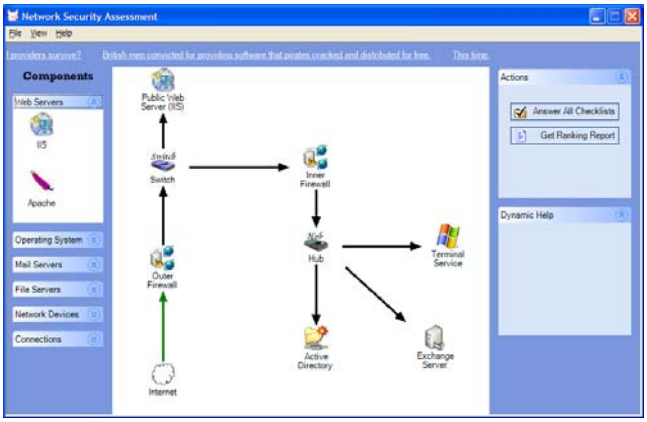

Figure 5: Drawing Area

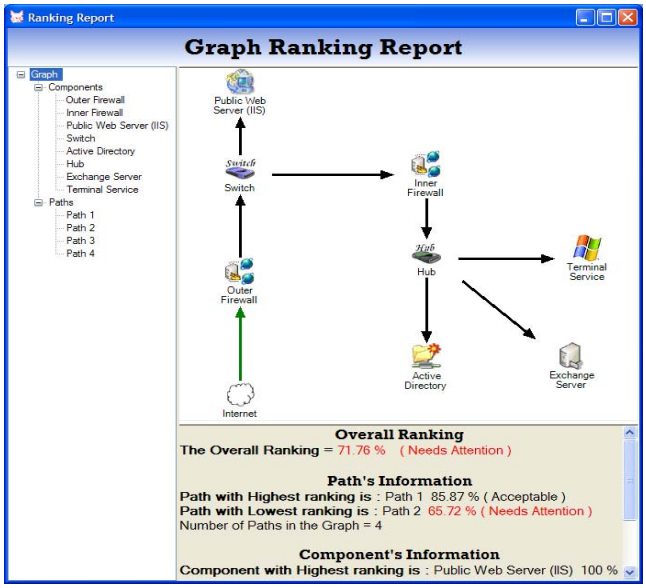

Figure 6: Security Ranking Report

4.2 Encrypting Information

As we mentioned earlier in section 4 that the saved file that contains the graph details and the answers for checklists for each components within the graph. This information is sensitive because it can contain the weakness points of the saved graph and exposing this information to anyone can compromise the network. That is why this information must be encrypted.

We faced some problems on where to store the encryption key. Storing the key in a file allows us to change the key easily if discovered but the problem with this approach is securing the file. Another option is putting the key hard coded within the code but the problem with this approach is the difficulty of changing the key if discovered.

The algorithm that we used ensures that each file has its own encryption key. When an administrator wishes to save his work, the program asks him to enter a password, and then from the password we generate a 128-bit key that is used to encrypt the information then store the ciphered data into the file. The key is generated using the `CryptDeriveKey` function from the .NET framework. After generating the key, we use it in the 3DES encryption algorithm. The idea behind the key generation algorithm is that the password seeds a cryptographically strong number generator that generates a cryptographically strong sequence of bytes that can be used as a key to the encryption algorithm. Before generating the key, the number generator is fed the password and a salt value to generate the key. The salt is a random series of

bytes. This can protect against dictionary attacks where the attackers derive a key for every word in the dictionary and try to use this list of pre-generated keys to decrypt the data. However, when using a non-zero salt value, the attacker needs to discover it and then regenerate the full key list for each document, which dramatically reduces the effectiveness of the dictionary attack [2] [3].

After the key is generated, it is sent to the algorithm to encrypt the document and save the cipher text in the file. When the administrator wants to open the file again, he is prompt to enter the password to regenerate the key again to use it to decrypt the file. If the password is wrong, a wrong key will be generated and the decryption will fail. If so, a message prompting the administrator that the password he entered is wrong [2].

By using this method, we do not worry any more about where to store the private key and whether it will be discovered or not. In addition, we can guarantee that every document generated with our tool will have a different key giving that they have different passwords.

4.3 Update Process

The security field is one of the fields that changes so frequently, so software designed to help improve security and do not support update get obsolete so fast. That is why we added the function to the tool.

To tool supports updating the security checklists and the components. They are updated by accessing a web server specially designed for the tool that contains the latest checklists and components [2].

Updating the checklists consists of adding new questions for old components, adding new checklists for new components, deleting questions and changing the questions. When we change the checklists on the web server, we update the date of that file, and when the tool looks for update, it compares the date of the checklist with the tool with the date of checklist on the web server and if web server's checklist is newer, it is downloaded. The same thing applies for updating the components [2].

5. Conclusion and Future Work

This paper summarizes what have we done to develop the tool and what function have we added to it. Of course, every work is hard to complete to the end. That is why we will mention the future work to be done in order to complete the tool.

We think that incorporating a tool such as GFI LAN Guard with our tool, then take advantage of the reports generated from that tool will increase the accuracy of our security report and will help the administrator to increase the security level of his network.

Another thing that needs to be added is to allow the administrator to add comments to each component within his network. We think that this will help him document his network such as adding information related to the place of the component, who is the administrator, contact information for the product company and so on.

One of the things that must also be improved is the ranking calculation formulas. We think that improving the formulas and calculation methods can help improve the security report.

As the tool goes through different kinds of improvements, we must not forget to ensure that the components and checklists are up-to-date because when releasing the latest version, the supported components must have the latest in the market.

Improving the drawing system can also result in a more elegant product. Such improvements include resizing the icons for the components, putting the copy, cut and paste functions, linking two graphs together to make it easy to draw and so on.

One important function that we feel is missing is the ability to print the security ranking report along with the graph in one printable form.

One last thing is to encrypt the checklists and components XML file for security reasons. This will prevent other users from using the checklists in other programs and steal our work.

6. REFERENCES

[1] Al-Abdulkareem M., *A Framework to Evaluate Web Information System*, to appear in The First International Conference on Safety and Security Engineering, June 2005, Rome, Italy.

[2] Al-Hokail, K and Al-Subai, S, *Network Security Assessment*, Graduation Project, King Saud University, Riyadh, 2004.

[3] Matthew MacDonald, *Microsoft Visual Basic .NET Programmer's Cookbook*, Redmond, Microsoft, 695.

[4] Phillips, Cynthia and Painton, Lura, *A Graph-Based System for Network-Vulnerability Analysis*, Proceedings of the 1998 workshop on New security paradigms, Charlottsville, VA, USA.,pp 71-79.

[5] Ammann, Paul, Wijesekera, Duminda and Kaushik, Saket, *Scalable, Graph-Based Network Vulnerability Analysis*, ACM Conference on Computers and Communication Security.

[6] Uribe, Tomas and Cheung, Steven, *Automatic Analysis of Firewall and Network Intrusion Detection System Configurations*, Proceedings of the 2004 ACM workshop on Formal Methods in Security Engineering.

[7] Guttman, Joshua, *Filtering Postures: Local Enforcement for Global Policies*, Proceedings of the 1997 IEEE Symposium on Security and Privacy.

[8] Jha, Somesh and Wing, Jeannette, *Survivability Analysis of Networked Systems*, Proceedings of the 23rd International Conference on Software Engineering, July 2001 Washington D.C., USA.