ATTACK GRAPH-BASED RISK ASSESSMENT AND OPTIMISATION APPROACH

Mohammed Alhomidi and Martin Reed

School of Computer Science and Electronic Engineering
University of Essex, Colchester, UK

ABSTRACT

Attack graphs are models that offer significant capabilities to analyse security in network systems. An attack graph allows the representation of vulnerabilities, exploits and conditions for each attack in a single unifying model. This paper proposes a methodology to explore the graph using a genetic algorithm (GA). Each attack path is considered as an independent attack scenario from the source of attack to the target. Many such paths form the individuals in the evolutionary GA solution. The population-based strategy of a GA provides a natural way of exploring a large number of possible attack paths to find the paths that are most important. Thus unlike many other optimisation solutions a range of solutions can be presented to a user of the methodology.

KEYWORDS

Attack Graph, Risk Assessment, Genetic Algorithm

1. INTRODUCTION

The need for security risk decision support models has become essential for a lot of organisations as well as network systems. Many network administrators and security analysts mostly rely on their skills and expertise rather than looking at objective metrics to support and justify the decision-making process. This paper aims to develop and provide a risk assessment framework that supports the process of making such decisions.

Security risk assessment is an essential process for managing risks in information systems for several reasons [1]: firstly, it helps organisations to quantify risks in their information systems. For example, risk assessment can produce numerical metrics that directly relate to monetary value for a specific threat or overall threat. Attack graph-based risk assessment can quantify risk for a single attack path from the source of attack to the target. It can calculate the likelihood of the attacker exploiting the target of attack as well as the impact as expected losses. Secondly, it supports decision makers to see and understand what risks their organisations may face. This helps decision makers to determine and make actions on risks which may be accepted, removed or mitigated. To illustrate this using attack graph-based risk assessment: the risks of all or some attack paths in the attack graph can be calculated so that decision makers can easily see what are the most and the least risky attack paths. Lastly, the attack graph evaluates control effectiveness after being implemented. For example, attack graph-based risk assessment can calculate the total loss in the whole attack graph before deploying controls and after deploying controls to determine the security saving of the organisation.

Risk Assessment methods can be classified as qualitative or quantitative [1]. Qualitative risk assessment produces descriptive estimates for risk such as “low,” “medium” and “high.” They are mostly based on traditional methods such as questionnaires, interviews and brainstorming etc. These methods are mainly used when organisations cannot estimate the likelihood and impact of threats. The methods have some drawbacks when being applied to very large complex network security environments [2]. On the other hand, quantitative risk assessment methods usually use predefined formulas and mathematical expressions to produce numeric estimates for risk, often measured in monetary value. According to a recent review paper on risk analysis methods in IT systems, most of the articles reviewed used quantitative risk methods [3]. In particular, quantitative methods are preferred in organisations which are able to provide estimates in numeric numbers for the probability and loss associated with each attack.

Although risk assessment is widely used to measure enterprise networks, it still cannot help organisations and decision makers in identifying, controlling and mitigating risks. According to Ou and Singhal, security risk analysis of networks faces some challenges [4]: first, managing the network security with hundreds of hosts and different operating systems and applications is difficult. Second, protecting the networks and critical systems from new evolving multi-step and multi-host attacks is tremendously hard. Last, detecting and preventing attacks using traditional detection methods is often not successful due to the complexity of attacks.

According to Gibson one of the most important steps for a risk management plan is to define the objectives [1]. These objectives are the indicators that show whether a risk management plan is successfully implemented or not. Gibson pointed out some common objectives for a risk management plan including: a list of threats, a cost-benefit analysis and security control costs. Many security decision support models are proposed or developed as a multi-objective problem, while others are developed as a single-objective problem. Developing a multi-objective risk assessment model can allow decision makers to consider a trade-off between objectives. Whereas, a single-objective risk assessment model usually gives decision makers a clear value for a single goal.

In this paper, we propose a quantitative attack graph-based risk assessment model to quantify the risks of attack paths in the attack graph. The risks of attack paths are calculated from the probabilities and expected losses associated with each vertex in the attack graph. The probability of vulnerability exploitation are considered as a numeric percentage taken from an expert knowledge database such as the Common Vulnerability Scoring System (CVSS)[5][6]. The expected loss is quantified in monetary terms.

The contribution of this paper is to develop a genetic algorithm approach to quantify the risk of attack paths in attack graphs. A population-based strategy is proposed to be a particularly useful approach as in an attack graph there are a very large number of possible paths. A single point optimisation strategy could be used to find the single most likely, or single most high-risk path. However, this may hide other very likely paths that might expose a risk. We use a quantitative metric towards single point optimisation goal, for example the highest risk of individual attack path, but the nature of a GA means that there will be many individuals available for final analysis. Thus a solution presented to the user will include many high-risk paths.

This paper is organised as follows: Section 2 discusses related works. In Section 3, we define an attack graph and provide examples of attack graphs and attack paths. Genetic algorithms are briefly explained in Section 3. Section 4 describes and explains the proposed security risk assessment model that includes vulnerabilities, attack likelihoods, expected losses and risks. Section 5 describes the proposed solution using a GA approach, which is then discussed using example experimental results Section 6.

2. Related Works

There are many works on security risk assessment which has its roots in general risk and failure analysis. An early work in risk assessment of computer systems is proposed by [7]. and presents a model to measure risks in networked systems. It defines the risk as the probability of consequences multiplied by the loss. It describes an optimisation approach to determine the best set of patched software running on a computer to minimise the risk. The work concluded that all software must be patched to reduce the risk. One shortcoming of this work is that it did not capture dependencies between exploits in terms of pre-conditions and post-conditions for network administrators to fully understand and properly analyse exploits. In this paper, we base the risk assessment model is on dependency attack graphs that represent the dependencies, relations and transition states between network configurations, vulnerabilities and exploits as attack paths.
Other security risk assessment models combine an attack graph with Bayesian networks [8] [9] [10] [11]. For example, Poolsappasit *et al*. developed a risk assessment model called a Bayesian attack graph [9]. The aim of the model is to assess security risk outcomes by calculating the expected loss and gain associated with every attribute in the Bayesian attack graph. However, this implicitly makes the assumption that the attacker has already compromised all states in the whole attack graph. In real-world scenarios, attackers do not necessary exploit all machines or vulnerabilities in a network (i.e. all states in the attack graph). In this paper, we calculate the risk of a range of attack graphs and present a range of the most and other highly risky attack paths.
Measuring security risks using attack graphs has been discussed in several works [11] [12] [13]. Noel and Jajodia developed a metric to measure the overall security of network systems [11]. In practice, the metric quantifies the risk through measuring the likelihood of an attacker compromising attack paths. The attack graph used in the work combines the exploits in attack paths with the initial condition. The authors assume that an attacker may start with a path and then follow another path. Therefore, both attack paths are measured in the metric. The model associates each network configuration with an implementation cost and reduces risk (expected loss from breach) by a certain amount. Lastly, the model finds the configurations that minimise the overall costs. Although this work has the capability to quantitatively analyse and optimise small attack graphs, it is challenging to use the technique to handle large graphs with thousands of network configurations. In this paper, we develop a genetic algorithm approach to handle very large graphs with thousands of vertices and edges.

3. Theoretical Background

3.1. Attack Graphs

Risk assessment requires an organisation to provide information about security threats. To be able to provide such information, the organisation should have a way or tool to represent and analyse security attacks. Attack graphs represent the combination of: hosts, network configurations, vulnerabilities and exploits to describe the possible known security attacks. In particular, Dependency Attack Graphs, as used in this paper, have the capabilities to show attack scenarios and paths between a source state and target state. Each vertex in a dependency attack graph represents a condition state of system settings, while the edge represents the casual relation between those system conditions [14] [15]. Each state in the dependency attack graph can be associated with numerical numbers to estimate the expected loss or the likelihood of the state being satisfied.

Definition 1: ***A dependency attack graph*** *can be specified as a directed acyclic graph G = (V, E, P, L) where V is a set of vertices that represent pre-conditions, vulnerabilities and exploits and E is a set of edges (arcs) that represent relationships between the pre-conditions, vulnerabilities* *and exploits. There is a probability P_i associated with each vertex that represents the likelihood of an attacker exploiting a vulnerability without considering the pre-conditions. There is an expected loss L_i associated with each vertex that represents the loss value in monetary units when the vertex has been exploited.*

Definition 2: *The **incoming** edge(s) of a vertex v from one or more vertices $W \subset V$ represents a dependency relationship such that the condition represented by v is dependent upon on W. Where v depends upon every condition in W then v is said to be an AND vertex; where v requires only one condition from W then v is said to be an OR vertex. The vertices in W are said to be parents of v.*

Figure 1 illustrates a very small attack graph to aid the description of the model, later realistic examples are too large to show the full picture. In this attack graph, the target of an attacker "vertex 1" depends upon some combination of the other vertices. In Figure 1, an AND vertex is represented by an ellipse and an OR is represented by a diamond. Thus vertex 1 can be exploited by either or both of the exploits represented by vertices 2 or 3. However, vertex 4 requires both vertices 6 and 7 as a joint pre-condition. Vertices 6, 7 and 8 are called LEAF vertices representing either a network configuration (e.g. an open port) or an existing vulnerability.

In the example of Figure 1 there are two ways that an attacker can reach the target: either with $1 \rightarrow 3 \rightarrow 4 \rightarrow 6$ AND 7 or $1 \rightarrow 2 \rightarrow 5 \rightarrow 7$ AND 8. While we may talk about a path a more accurate description would be to describe them as a tree and thus we define.

Definition 3: *A minimal attack tree $T \subset G$ is an attack tree defined by a set $M \subset V$ vertices and a set $N \subset E$ edges. The attack tree starts with a set $L \in M$ LEAF vertices denoted as the source vertices of the attack, such that the in-degree of $l \in L$ is 0 and ends with a target vertex $t \in M$ as the target of attack, such that the out-degree of the target vertex t is 0. Each OR vertex has only one parent vertices $w \subset M$ whereas each AND vertex t has every parent vertex $s \subset V$.*

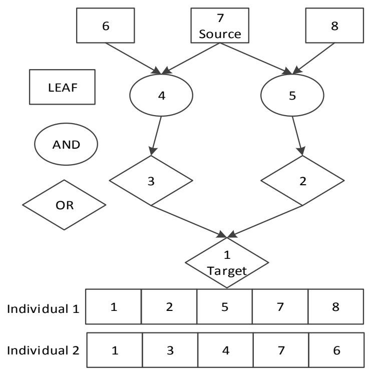

Figure 1. Attack tree example

Thus in the example shown in Figure 1, the node set {1,3,5,7,8} represents one minimal attack tree and the set {1,2,5,7,8} represents the other possible minimal attack tree.

3.2 Genetic Algorithms

The concept of genetic algorithms was first introduced by Holland and his student Kenneth [16] [17]. The genetic algorithm encodes a specific problem on a chromosome-like data structure. It applies genetic operators to explore the solution space and produce potential solutions. The GA typically has: a population of chromosomes, selected according to fitness; crossover to produce new offspring population; and, mutation of new offspring.

In our risk assessment model, we use the GA approach for several reasons: first, the GA produces populations of solutions not just a single solution, an operator is rarely interested in a single attack vector at attackers may choose many paths (or trees in out representation); second, it is highly suited to NP-complete problems such as searching through many promising attack paths/trees; lastly, the GA can encode many objectives and calculate a large range of possible risk values so that they can be used for decision making.

4. PROPOSED RISK ASSESSMENT MODEL

This paper describes the risk assessment framework shown in Figure 2. First, Attack graph generation scans the network using a network scanner i.e. Nessus [18] and then generates dependency attack graphs using MulVAL [19]. Next, likelihood determination assigns a probability to each vertex. Third, loss estimation quantifies loss for each vertex. Later, risk determination calculates the risk associated with vertex. Fifth, GA optimisation calculates the risk of attack paths in the attack graph. Last, high-risk attack paths are presented to the user by the GA.

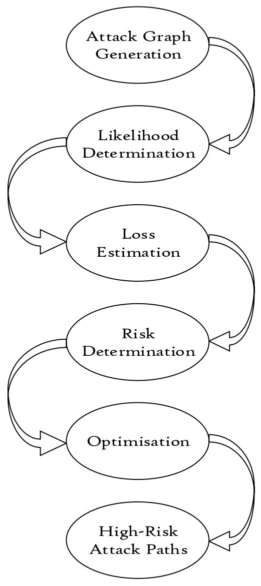

Figure 2. Risk assessment and optimisation framework

4.1 Likelihood Determination

Most risk assessment models contain at least the probability of attack or threat. In the previous section, we describe three types of vertices in the attack graph. One of the types is the Leaf vertex that represents an existing vulnerability, a network configuration or condition. We assign the probability of vulnerability to leaf vertices according to the CVSS scores [5] [6]. The main purpose of CVSS is to provide an open source framework for the characteristics and impacts of IT vulnerabilities. Therefore, it has been widely used to derive probabilities of vulnerabilities. Each CVSS score addresses three areas of concerns termed [6]: base, temporal and environmental. Each metric produces a numeric number in scale of (1 to 10). The base group provides quantities of vulnerability. The temporal group reflects the characteristics of vulnerability that change over time. The environmental group describes a specific vulnerability that is unique to any user's environment. CVSS is used in this paper because it offers standardised vulnerability scores. The scores cover all software and hardware platforms.

Leaf vertices representing network configurations, or conditions, are assigned a probability of 1.0 because they are always assumed to be true. For example, a network configuration can be an open TCP port 80. To assign a probability to this condition only makes sense in the context that it is either true or false: if the TCP port 80 is open, it indicates a probability of 1.0; else, the TCP port 80 is not open indicating the probability of 0.

AND vertices representing MulVAL rules are assigned scores according to the MulVAL Reasoning System [19]. OR vertices representing exploits are assigned probabilities according to the incoming vertices.

4.2 Loss Estimation

Loss estimation has always been a challenge in quantitative risk assessment often due to the lack of internal or external data [1]. The loss estimation value assesses the loss when an attack actually occurs. For instance, a web server running many services for an enterprise company is unavailable for a day, the company will have a severe impact because the business services cannot be provided any more. At this point, estimating the loss may include the hourly revenue of business, loss of data and implementing a mitigation plan. There are a few types of loss that companies experience as a results of a security breach [20]:

- Productivity loss is the time, e.g. hours or days, that a service or application is not available. For example, the downtime lost due a security attack and therefore the loss can be estimated as the average hourly revenue of the system or service multiplied by the number of hours that the system is unavailable.
- Data loss is the corruption, copying or unauthorised access of data. For instance, data loss happens in the event of unauthorised access to sensitive data of an enterprise company. The loss estimation can be the value of money the company will pay to recover the sensitive data.
- Business loss is concerned with the reputational impact. An example is that the customers of a business many no longer use the business because of highly data breach. In this case, it is difficult to quantify the loss due to several factors such as publicity and severity of security breach. However, estimates can be made from looking at similar examples and the impacts they cased through loss of custom or share values changes.

The estimation of loss in our work is quantified as one monetary value which should include all types of losses. In this work, determining accurate loss estimates remains a key issue, but we aim to develop a genetic risk assessment model that can be applied to any organisation. To successfully apply the model, the organisation is the only one that can produce accurate loss estimates. Each enterprise has its own organisational architecture, operational procedures, services and assets. In this case, the enterprise CEO and system administrators are the only people who are able to produce some estimations of losses. Therefore, for evaluation of techniques, it is common to produce loss estimates using theoretical aspects such as statistical distributions, which can be used to simulate real data. Operational Risk Management research has investigated the risk of directed and indirect losses resulting from inadequate or failed internal processes, people, and systems, or from external events such as security breaches. Several research studies [21] [22] [23] [24], have reported that the lognormal distribution is often the best to fit the severity distribution of the monetary loss data, and this distribution is used in this work.

It is shown in Figure 1 that the attack graph has three types of vertices "AND, OR, LEAF", where the OR vertex represents an exploit. In this work, OR vertices are associated with lognormal distributed values to represent loss; an OR vertex does not represent any loss unless an exploit occurs i.e. when an OR vertex is satisfied in the attack graph. However, AND/LEAF vertices are associated with zero losses because they do not indicate any direct attack or exploited threat but rather AND vertices link possible exploits represented in the OR vertices.

4.3 Risk

The risk is represented as the product of the cumulative probability of a vertex v and the expected loss.

Definition 4: *The risk function of a vertex v ∈ V is defined as the product of the cumulative probability of v with the expected loss of v.*

$$R(T_i) = P(v) * L(v) \tag{1}$$

where *P*(*v*) is the cumulative probability that represents the likelihood that a vulnerability associated with vertex is exploited. *L*(*v*)is the expected loss in monetary value associated with a vertex v if vertex v is exploited.

Definition 5: *The risk of a minimal attack tree T is the sum of all risk values.*

$$\max \sum_{v \in V} R(T_i) \quad \forall T_i \in T \tag{2}$$

Note that each vertex v in Ti has a probability

$$0 < P(v) \leq 1 \, \forall v \in V$$

and each vertex v in Ti has an expected loss

$$0 \leq L(v) \quad \forall v \in V$$

5. PROPOSED GENETIC ALGORITHM

Algorithm 1 describes a high level view of the proposed GA methodology for attack graph-based risk assessment and optimisation. The optimisation aim of the GA is to find the highest risk minimal attack tree. It should be noted that although this is the optimisation target of the GA, the aim of the whole process is to find a population of high-risk paths. The inputs are the attack graph G which consists of vertices V, edges E, probabilities P and losses L. The attack path is a set of vertices that represents an attack scenario from the source to the target of the attack. The probabilities associated with each vertex in the attack path is actually the probability of vertex being satisfied. Similarly, the expected losses associated to each vertex in the attack path is the actual loss in monetary value if the vertex is exploited.

The GA works as the following: In step 1, the GA randomly generates an initial population $\mathbb{P}$. with the given size α. Each individual member, as shown in Figure 1, is an integer number that represents the vertex's ID and indicates whether this vertex is actually included in the attack path or not. Step 2 evaluates each individual in the population $\mathbb{P}$ and then initialises a list of *BestSolutions* in step 3. Step 4 begins the main while loop of the GA which continues until the stopping condition is met i.e. the best solution does not improve within 200 generations. Steps 5-

11 repeat to produce *offspring* and stops when the number of offspring is equal to α. Step 6 randomly selects a pair of parents for reproduction. Step 7 performs the crossover process based on the rate of crossover γ. Step 8 performs the mutation process based on the rate of mutation μ. Step 9 evaluates the produced offspring by computing its fitness. Step 10 adds the produced offspring to the offspring population $\mathbb{P}'$. Step 11 updates the *BestSolutions* by checking if the

```
Inputs: $G = (V, E, P, L)$, $\alpha$ is the population size,
$\gamma$ is the rate of crossover and $\mu$ is the rate of mutation.
Output: BestSolution (High-risk attack paths).
1. $\mathbb{P} \leftarrow$ Generate $\alpha$ feasible solutions
2. Evaluate each individual $\in \mathbb{P}$
3. Initialise BestSolutions
4. While stopping condition is not met
5. Repeat until $\alpha$ is reached
6. Select a pair parents from $\mathbb{P}$
7. Apply crossover according to $\gamma$
8. Apply mutation according to $\mu$
9. Evaluate offspring
10.        Add offspring to offspring population $\mathbb{P}'$
11.        Update BestSolutions
12. Replace current $\mathbb{P}$ with $(\mathbb{P} \cup \mathbb{P}')$
13. End while
```

Figure 3. Pseudo-code of proposed GA

produced offspring is better than other best solutions. Step 12 preforms the population replacement by combining the current population $\mathbb{P}$ and the offspring population $\mathbb{P}'$ but keeping the fittest individuals for the next generation.

6. EXPERIMENTAL EVALUATION

6.1 Network Configuration

Figure 4 shows a more realistic network topology that includes four machines. Two machines are database servers (Host 1 and 2), one web server (Host 3) and one desktop (Host 4). The list of vulnerabilities for each host is summarised in Table 1. The categories of vulnerabilities include Critical, High, Medium, Low and Information according to severity. For example, Host 1 has 14 critical, 35 high and 95 medium vulnerabilities.

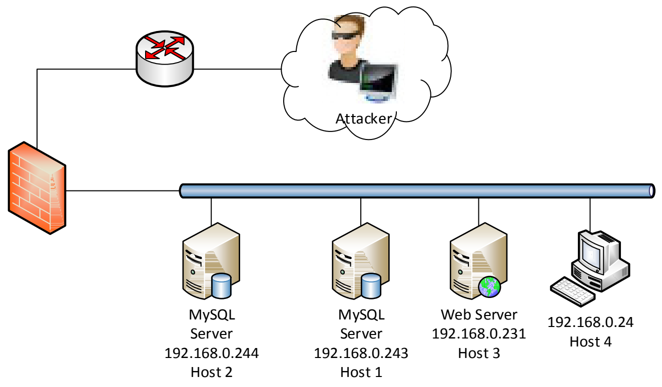
Attacker
MySQL Server 192.168.0.244 Host 2
MySQL Server 192.168.0.243 Host 1
Web Server 192.168.0.231 Host 3
192.168.0.24 Host 4

Figure 4. Real-world network

Table 1. Summary of number and types of vulnerabilities of each host

Host	Critical	High	Medium	Low	Info	Total
1	14	35	95	13	144	301
2	14	33	89	11	126	273
3	3	17	62	6	68	156
4	0	0	4	1	32	37

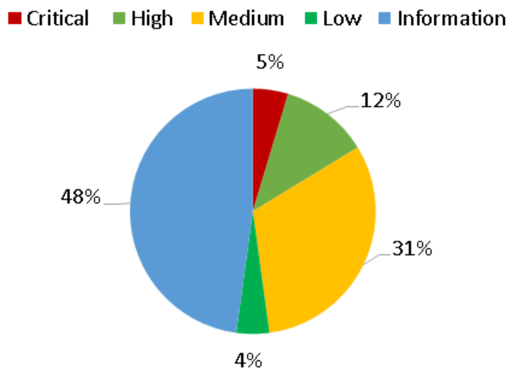

Figure 5. Summary in percentage of identified vulnerabilities

Figure 5 shows another way to describe vulnerabilities in the network according to the categories. It is seen that the critical vulnerabilities are 5% of the total number of vulnerabilities. The high is 12%, medium is 31%, low 4% and information is 48%.

6.2 Results

Following the process described in Section 4, the attack graph is generated using MulVAL. The attack graph consists of 1682 vertices and 3183 edges. We have explored many different GA parameters to determine the best parameters that produce good solutions in short run times. Table 2 lists results from some of the experiments performed to explore the average fitness and average run-time averaged over ten GA runs. It is seen that the average fitness results are similar but the average run-times differ widely. The highest average fitness is found with the population size

4500, the rate of crossover 0.95 and the rate of mutation 0.3. The average run time seems to increase as the population size increases because the GA takes more time to produce and process more individuals.

Figure 6 shows the GA results with population size 500 and different crossover and mutation probabilities in bar charts. On the left side of Figure 6, the average fitness for the GA experiments seem to be similar. However, the left side shows the average run time in (ms) and indicates that the higher crossover rate 0.95 and mutation rate 0.5 took the largest run time over 10000 ms. Similarly, Figure 7 shows the GA results with population size 2000.

Table 2. An extract of GA experimental results exploring the population size (PS), rate of crossover (Pc), rate of mutation (Pm), average fitness (AvgFit) and average time (ms)

PS	Pc	Pm	AvgFit	AvgTime(ms)
200	0.95	0.3	3312	3403
500	0.95	0.001	3119	3565
500	0.95	0.1	3320	6975
500	0.95	0.3	3394	9161
1000	0.8	0.001	3167	9314
1000	0.8	0.1	3326	14884
1000	0.8	0.3	3445	17976
1000	0.95	0.3	3513	17069
2000	0.95	0.001	3319	20436
2000	0.95	0.1	3469	29433
2000	0.95	0.3	3519	34785
3000	0.8	0.001	3415	34734
3000	0.95	0.001	3422	38316
3000	0.95	0.1	3558	42103
3000	0.95	0.3	3614	54686
4000	0.95	0.3	3552	77797
4500	0.95	0.3	3646	80908
5000	0.95	0.3	3627	100896
5500	0.95	0.3	3638	91219
6000	0.95	0.3	3637	120671

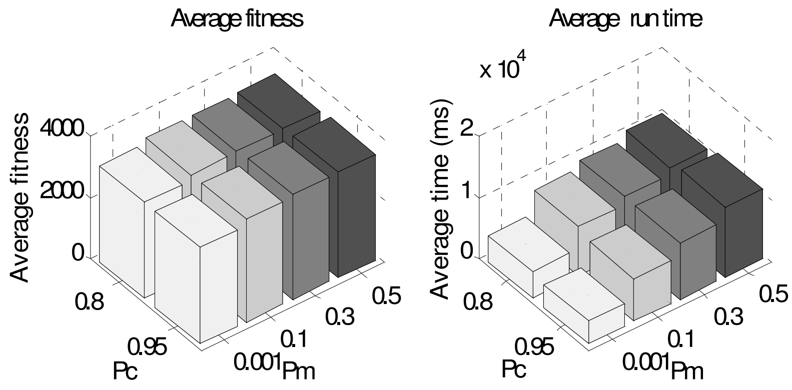

Figure 6. GA results with population size 500

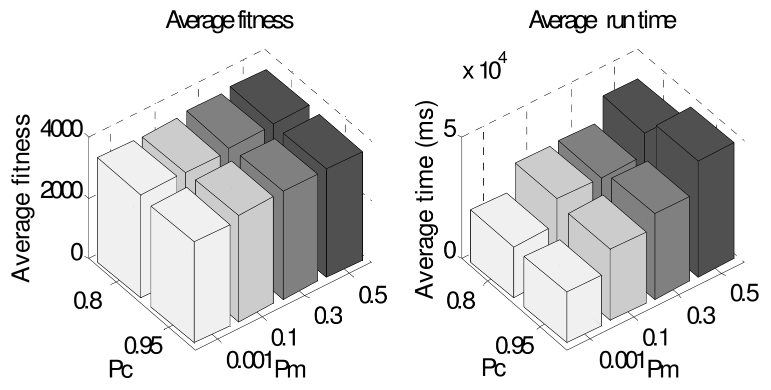

Figure 7. GA results with population size 2000

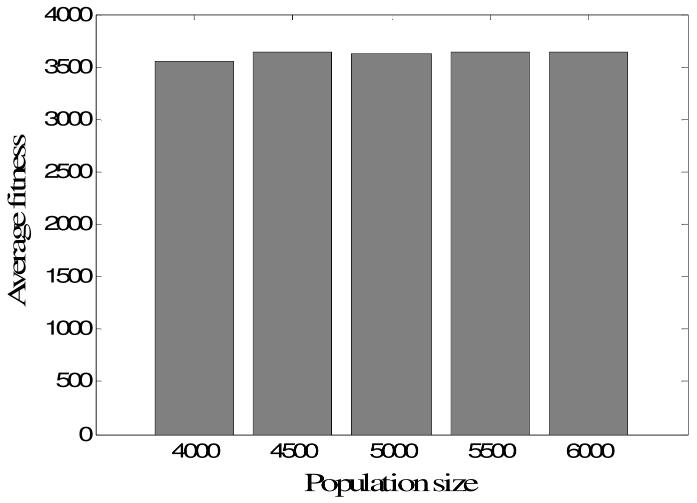

Figure 8. GA results showing average fitness vs. population size

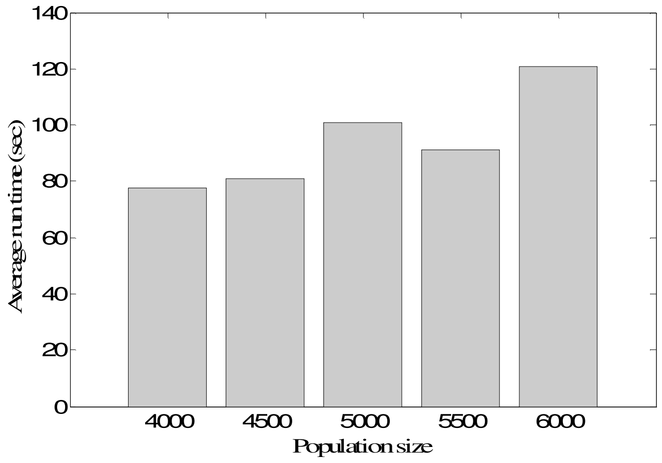

Figure 9. GA average run time (sec) for several population sizes

Figure 8 presents average fitness with different population sizes. The bars show that the results are around 3500. Figure 9 shows the GA average run time in seconds for several population sizes.

7. CONCLUSIONS

This paper presents a risk assessment model based on attack graphs. The work demonstrates that attack graphs are very useful tools in risk assessment. More importantly, they can be used to quantitatively measure and explore the risk of networked systems. We develop a GA methodology to analyse attack graphs, compute risk values of attack paths and produce necessary risk information for networks. The GA approach makes it possible to handle very large attack graphs. Additionally it has the strong advantage that the GA generates populations of solutions from which the overall risk from a number of individual attack paths can be quantified. The population can be presented to a user for expert analysis of the highest risk attacks.

REFERENCES

[1] Gibson, D., (2010) “Managing Risk in Information Systems”, Jones & Bartlett Learning.

[2] ISO/IEC13335-2001, (2001) “Information Technology-guideline for the Management of IT Security”, The International Organization for Standardization.

[3] Sulaman, S. M., Weyns, K. and M. H, (2013) “A review of research on risk analysis methods for IT systems” presented at the *Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering*, Porto de Galinhas, Brazil.

[4] Ou, X. and Singhal, A. (2011) “Quantitative security risk assessment of enterprise networks” Springer.

[5] Peter, M., Karen, S. and R. Sasha, (2006) “Common Vulnerability Scoring System” *Security & Privacy*, IEEE, vol. 4, pp. 85-89.

[6] Mell, P., Scarfone, K. and Romanosky, S., (2007) “A complete guide to the common vulnerability scoring system version 2.0”, pp. 1-23.

[7] Bilar, D., (2003) “Quantitative risk analysis of computer networks” Dartmouth College, 2003.

[8] Frigault, M., Wang, L., Singhal, A., and Jajodia, S.,(2008) “Measuring network security using dynamic bayesian network”, pp. 23-30.

[9] Poolsappasit, N., Dewri, R. and Ray, I., (2012) "Dynamic Security Risk Management Using Bayesian Attack Graphs" *Dependable and Secure Computing, IEEE Transactions on*, vol. 9, pp. 61-74.

[10] Peng, X. , Li, J. H., Xinming, O., Peng, L. and Levy, R., (2010) "Using Bayesian networks for cyber security analysis," in *Dependable Systems and Networks (DSN)*, IEEE/IFIP International Conference on, pp. 211-220.

[11] Noel, S., Jajodia, S., Wang, L. and Singhal, A., (2010) "Measuring security risk of networks using attack graphs" vol. Vol. 1..

[12] Wang, L., Singhal, A. and Jajodia, S., (2007) "Measuring the Overall Security of Network Configurations Using Attack Graphs" *Data and Applications Security XXI*, vol. 4602, pp. 98-112.

[13] Chen, F. and Su, J.-S., (2008) "A Flexible Approach to Measuring Network Security Using Attack Graphs" in *Electronic Commerce and Security International Symposium*, Guangzhou, China, pp. 426-431.

[14] Ou, X., Boyer, W. F. and McQueen, M. A., (2006) "A scalable approach to attack graph generation" presented at the *Proceedings of the 13th ACM conference on Computer and communications security*, Alexandria, USA.

[15] Noel, S. and Jajodia, S., (2004) "Managing attack graph complexity through visual hierarchical aggregation" in *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, Washington DC, USA, pp. 109-118.

[16] Holland, J., (1975) "Adaptation in Natural and Artificial Systems," University of Michigan Press, Ann Arbor.

[17] Goldberg, D. E. and Holland, J. H., (1988) "Genetic algorithms and machine learning," Machine Learning, vol. 3, pp. 95-99.

[18] Tenable Network Security, (Accessed Date 11 February 2013) "Nessus Vulnerability Scanner" Available: http://www.tenable.com/products/nessus/nessus-product-overview

[19] Ou, X., Govindavajhala, S. and Appel, A. W., (2005) "MulVAL: A logic-based network security analyzer" in *14th USENIX Security Symposium*, pp. 8-8.

[20] Purcell. J., (2009) "Quantifying Business Value of Information Security Project".

[21] Käärik, M. and Zegulova, A., (2012) "On estimation of loss distributions and risk measures" Acta et Commentationes Universitatis Tartuensis de Mathematica, vol. 16.

[22] Êlvarez, G., (2001) "Operational Risk Quantification".

[23] Finke, G. R., Singh, M. and Rachev, S. T., (2010) "Operational risk quantification: a risk flow approach" *Journal of Operational Risk*, p. 65, 2010.

[24] De Fontnouvelle, P., DeJesus-Rueff, V., Jordan, J. and Rosengren, E., (2003) "Using loss data to quantify operational risk" Federal Reserve Bank of Boston Working Paper, pp. 03-5.