Research Journal of Applied Sciences 10 (11): 758-762, 2015
ISSN: 1815-932X

Minimization Analysis of Network Attack Graphs Using Memetic Algorithm

[1]Azam Faraji and [2]Mohammad Ebrahim Shiri Ahamd Abadi
[1]Department of Computer, Borujerd Branch, Islamic Azad University, Borujerd, Iran
[2]Department of Computer Science, Faculty of Mathematics and Computer Science,
Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran

Abstract: As computer networks continue to grow, it becomes increasingly more important to automate the process of evaluating their vulnerability towards attacks. Despite the best efforts of software architects and developers, network hosts inevitably contain a number of vulnerabilities. Attack graphs are models that offer significant capabilities to analyze security in network systems. An attack graph allows the representation of vulnerabilities. We model compositions of vulnerabilities through attack graphs. This study proposes a Memetic based method to explore the graph attack. 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 Memetic solution. The population-based strategy of a Memetic provides a natural way of exploring a large number of possible attack paths to find the paths that are most important. Simulated annealing is used as local optimizer in proposed Memetic algorithm. A comparison was made between presented algorithm and Memetic Particle Swarm Optimization algorithm. Experimental results proved that Memetic based algorithm for minimization analysis of network attack graph is accurate and has better performance.

Key words: Attack graph, Memetic algorithm, vulnerabilities, simulated annealing, computer networks

INTRODUCTION

Our society has become increasingly dependent on computer networks and the trend towards larger networks will continue. Each network host runs different software packages and supports several modes of connectivity. Despite the best efforts of software architects and developers, network hosts inevitably contain a number of vulnerabilities. Hence, it is not feasible for a network administrator to remove all vulnerabilities present in the network hosts. Therefore, the recent focus in security of such networks is on analysis of vulnerabilities globally, finding exploits that are more critical and preventing them to thwart an intruder.

Researchers and penetration testers often organize these chains of exploits into graphs. Such a graph will not be very useful if it cannot inform a system administrator with detailed information about the discovered problems.

Phillips and Swiler proposed the concept of attack graphs where each node represents a possible attack state. Edges represent a change of state caused by a single action taken by the intruder. In particular, an attack graph that illustrates all possible multi-stage, multi-host attack paths is crucial for a system administrator to understand the nature of the threats and decide upon appropriate countermeasures. Attack graphs are used to determine if designated goal states can be reached by attackers attempting to penetrate computer networks from initial starting states. For this use, they are graphs in which the starting node represents an attacker at a specified network location. Nodes and arcs represent actions the attacker takes and changes in the network state caused by these actions. Actions typically involve exploits or exploit steps that take advantage of vulnerabilities in software or protocols. The goal of these actions is for the attacker to obtain normally restricted privileges on one or more target hosts, where the target could be a user's computer, a router, a firewall or some other network component. Many actions that compromise separate hosts and use them as stepping stones may be required in large attack graphs to reach the target host.

A full attack graph will show all possible sequences of attacker actions that eventually lead to the desired level of privilege on the target. As networks of hosts continue to grow in size and complexity, evaluating their vulnerability to attack become increasingly more important to automate. This makes the generation of attack graph become a classical problem for network security analysis. Various kinds of attack graphs have been proposed for analyzing network security. Phillips and Swiler developed a tool for generating attack graph. The tool constructs the attack graph by forward

Corresponding Author: Azam Faraji, Department of Computer, Borujerd Branch, Islamic Azad University, Borujerd, Iran

exploration starting from initial state using attacker's profile, configuration information of networked host and a database containing template of action. Attack templates represent generic attacks including pre and post conditions uch as operating system version which must hold for the attack to be possible and the privilege obtained after the attack is over. The configuration file gives detailed information about the specific system to be analyzed and the assumed attacker's capabilities.

LITERATURE REVIEW

Ou presented logical attack graphs which directly illustrate logical dependencies among attack goals and configuration information. Their attack graph generation tool builds upon MulVAL (Sheyner *et al*., 2002), a network security analyzer based on logical programming. The aim of minimization analysis of network attack graphs is to find a minimum critical set of exploits that completely disconnect the initial nodes and the goal nodes of the graph.

Sheyner *et al*. (2002) showed this problem is in fact NP-hard. They proposed an approximation algorithm, ApproxNAG which can find an approximately-optimal set of exploits which must be prevented to thwart an intruder.

Noel *et al*. (2010) used attack graph for measuring security risk of networks. Proposed attack graph metric quantifies this risk through measuring the likelihood that such residual paths may eventually be realized by attackers. When a network is more secure, attack likelihood is reduced. Abadi and Jalili (2008) presented an Ant Colony Optimization algorithm, AntNAG and a Genetic algorithm, GenNAG, for minimization analysis of network attack graphs.

Alhomidi and Reed (2014) introduced a Genetic algorithm to explore graph attacks. 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.

Abadi and Jalili presented a binary particle Swarm Optimization algorithm with a time-varying velocity clamping, called SwarmCAG-TVVC, for minimization analysis of cost-sensitive attack graphs. The aim of study is to find a critical set of countermeasures with minimum weight whose implementation causes the initial nodes and the goal nodes of the graph to be completely disconnected.

Another approach is to embed a local optimizer in between the iterations of the global search heuristics. By doing this, exploration and exploitation occur in parallel (Engelbrecht, 2005). Such hybrids of local and global search heuristics have been studied elaborately in the evolutionary computation paradigm (Eiben and Smith, 2003) are generally referred to as Memetic algorithms (Krasnogor *et al*., 2006).

While Evolutionary algorithms take inspiration from biological evolution, Memetic algorithms mimic cultural evolution. The term meme refers to a unit of cultural information that can be transmitted from one mind to another after reinterpretation and improvement that in the context of combinatorial optimization corresponds to local search.

ParticleNAG is a Memetic PSO algorithm for minimization analysis of large-scale network attack graphs. A performance comparison was made between ParticleNAG algorithm and ApproxNAG, AntNAG and GenNAG. The reported results proved that ParticleNAG has better performance than the rest.

In this study, a Memetic based minimization analysis of network attack graph is proposed. Also a performance comparison was made between Memetic based algorithm and ParticleNAG.

MEMETIC ALGORITHM

Memetic Algorithm (MA) (Moscato and Cotta, 2007) has been widely used in recent years, mainly due to their success in solving many hard optimization problems, attracting experienced researchers to work on the challenges of this field. Memetic Algorithms (MAs) are a class of stochastic global search heuristics in which Evolutionary algorithms-based approaches are combined with problem-specific solvers. The later might be implemented as local search heuristics techniques, approximation algorithms or sometimes, even (partial) exact methods. In MA, hybridization is done to either accelerate the discovery of good solutions for which evolution alone would take too long to discover or to reach solutions that would otherwise be unreachable by evolution or a local method alone. Large majority of Memetic algorithms use heuristic local searches rather than, e.g., approximation or exact methods. There are several reasons why it is worthwhile hybridising Evolutionary algorithms with local searchers, among them we could mention:

- (Sub) problem specific information can be distilled into variation operators, e.g., crossover and mutation or into local searchers as to effectively bias the search process towards promising regions of the search space
- Better diversity and robustness for finding optimized solutions

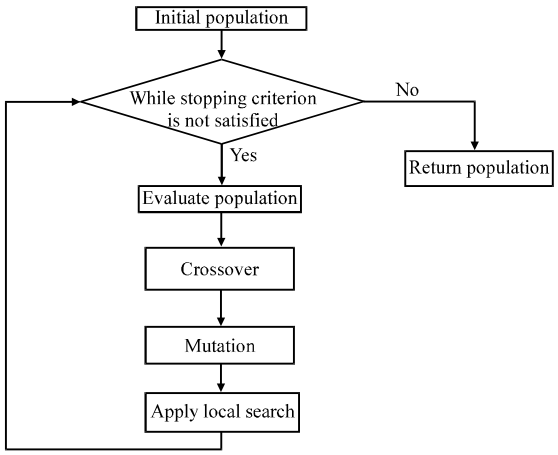

Fig. 1: Flowchart of Memetic algorithm

Memetic algorithms have received various names throughout the literature and scientist not always agree what is and what is not an MA due to the large variety of implementations available. Some of the alternative names used for this search framework are hybrid GAs, Baldwinian EAs, Lamarckian EAs, genetic local search algorithms, etc. The term MA is now widely used as a synergy of evolutionary or any population-based approach with separate individual learning or local improvement procedures for problem search.

Like Evolutionary algorithms, MAs are population-based metaheuristics. Each individual or agent represents a tentative solution for the problem under consideration. These solutions are subject to processes of competition and mutual cooperation in a way that resembles the behavioral patterns of living beings from a same species. Figure 1 shows flowchart of Memetic algorithm.

PROPOSED ALGORITHM

Since security analysis mainly aims at current computer network, it needs a simple flexible and complete model to reduce complexity of system states space. Through analyze a simplified network security model and determine whether the network security requirements were met. We use a State-based Model (TCP/IP Model) of network security to discover attacks paths.

The devices on the network are the basic elements of information system, for example, computers, routers, switches.

We use a set, H = {h1, h2 ... hm} to represent these devices and hi (i = 1, 2, ..., m) represents a single network device. A host on the network is represented by a triple (HOSTID, OS, PLVL, VULS). HOSTID is the unique identifier of host on the network, it can be the IP address or host name. OS is the type and version of operation system. PLVL is the level of privilege that the intruder has on the host and VULS is a set of host-specific vulnerable components. Three privileges are defined in network:

- ROOT: system administrator, managing all system resources
- USER: any general system user which is created by administrator
- None

Each exploit e∈E is a tuple (PRE, HOST, DEST, POST) where PRE is a list of conditions that must hold before launching the exploit, HOST is the host from which the exploit is launched, DEST is the host targeted by the exploit and POST specifies the effects of exploit on the network. An exploit e∈E is inevitable if its prevention is not feasible or incurs high cost. To prevent an exploit e∈E, the security analyst must implement a suitable countermeasure such as:

- Changing the firewall configuration
- Patching the vulnerability that made this exploit possible
- Deploying a host-based or network-based intrusion detection and prevention system
- Modifying the configuration of network services and applications
- Deleting user accounts
- Changing access rights
- Setting up a Virtual Private Network (VPN)

We used a system which is depicted in Fig. 2 for minimization analysis of network attack graphs. Vulnerability scanning tools such as Nessus Deraison or Lufeng *et al.* (2009), determine vulnerabilities of individual hosts. Using this vulnerability information along with exploit templates, intruder's goals and other information about the network such as connectivity between hosts, a network attack graph is generated. In this directed graph, each complete path from an initial node to a goal node corresponds to an attack scenario. The minimization analysis of the network attack graph determines a minimum critical set of exploits that must be prevented and countermeasures that must be implemented which leads to situation with guarantee of no attack scenario is possible.

The aim of minimization analysis of network attack graphs is to find a minimum critical set of exploits/ countermeasures. In this study, A Memetic based algorithm for minimization analysis of network attack graphs is proposed. Let E = {e_1, e_2, ..., e_n} be the set of preventable exploits. Each chromosome corresponds to an

```
s≥S, e<- E(s), Temp = T
While (e>emin)
    Snew≤neighbor (s)
    enew≤E (snew)
    If (P (e, enew, Temp)>random ())
        s≥snew, e≥enew
    Temp = Temp-coolingRate
```

Fig. 2: Pseudo code of simulated annealing in proposed Memetic algorithm

n-bit vector (g_{i1}, g_{i2}, ..., g_{in}) and represents a subset of exploits and represents a subset of exploit ($E_i \in E$) in which the exploit $e_j \in E_i$ if and only the element $g_{ij} = 1$:

$$E_i = \{e_j \in E | g_{i,j} = 1\}$$

Equation 1 calculates the sum of exploits in chromosome (i) in proposed algorithm:

$$W(i) = \sum_{e_j \in E_i} w(e_j) \quad (1)$$

The algorithm evaluates chromosomes with value of W. Let $S = \{S_1, S_2, ..., S_n\}$ be the set of attack scenarios represented by the network attack graph G. The attack scenario $S_k \in S$ is hit by chromosome (i) if:

$$S_k \cap E_i \neq \varnothing$$

The chromosome (i) represents a critical set of exploits if all attack scenarios are hit by it. If a chromosome fails to satisfy all scenarios, it will be improved by Local Search algorithm. Simulated annealing is used in this study as local optimizer.

In local optimizer part of proposed Memetic algorithm a new neighbor state will be generated by changing troublesome genes with randomly chosen items. Improvement continues until minimum temperature reaches or new chromosome (neighbor state) satisfies all scenarios. Figure 2 shows pseudo code of simulated annealing in proposed Memetic algorithm. Pseudo code of Memetic based for minimization analysis of network attack graph is as given:

$$p \begin{cases} 1 & \text{currentEnergy<newEnergy} \\ e^{\frac{\text{newEnergy-currentEnergy}}{\text{Temperature}}} & \text{currentEnergy>newEnergy} \end{cases} \quad (2)$$

```
Initialize popultion
While (generation<MaxGeneration)do
    Evaluate each chromosome
    Apply crossover
    Apply mutation
    Apply local optimizer
Endwhile
Report final soluation
```

Fig. 3: Pseudo code of proposed Memetic based algorithm for minimization analysis of network attack graph

Also Fig. 3 mentions to pseudo code of proposed Memetic based algorithm for minimization analysis of network attack graph.

Single point method is used for cross over operation. A point will be selected on both parents and all data beyond that point will be swapped between the two parents. Flip Bit Method is used for Mutation operation as well.

EXPERIMENTAL RESULT

In order to test the performance of our approach, we constructed several testing models based on networks of varying sizes and complexity. In Table 1, five weighted attack graphs that have been randomly generated and used for testing Proposed algorithm are shown.

A comparison was made between Proposed algorithm and Memetic Particle Swarm Optimization algorithm. Table 2 reports the comparison of execution times in these two algorithms. Parameters of proposed Memetic algorithm were initialized as follow:

- Number of generation: 10
- Probability of crossover: 0.05
- Probability of mutation: 0.05
- Local optimizer: simulated annealing

Figure 4 shows effects of the chromosome size on the performance of proposed algorithm in the experiments for minimization analysis.

Table 1: Weighted attack graphs

Attack graphs	Scenarios	Sum of weights	Average weight in scenario
Attack graph 1	8	56	7
Attack graph 2	15	150	10
Attack graph 3	20	300	15
Attack graph 4	25	400	16

Table 2: Comparison between proposed Memetic algorithm and Memetic Particle Swarm Optimization algorithm

Algorithm	Attack graph	Execution time (sec)
Memetic	1	1.600
Memetic Particle	1	7.260
Memetic	2	10.250
Memetic Particle	2	31.470
Memetic	3	16.369
Memetic Particle	3	48.090
Memetic	4	23.580
Memetic Particle	4	76.350

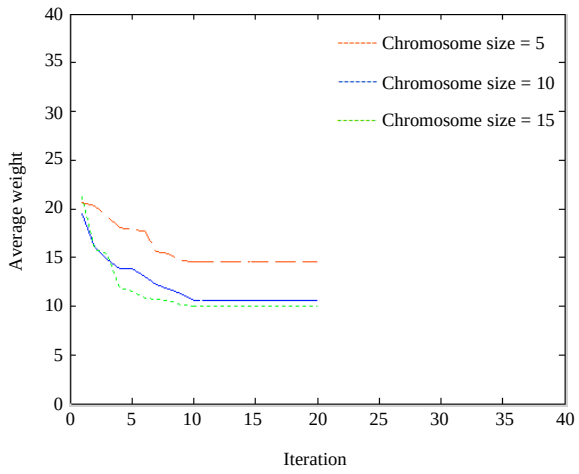

Fig. 4: Effects of the chromosome size on the performance of proposed algorithm

CONCLUSION

Analysis of attack graphs is considered as a time consuming and complex problem. Numerous approaches have been proposed such as genetic algorithm, Ant Colony Optimization algorithm and Particle Swarm Optimization algorithm but most of them failed to satisfy time complexity and accuracy which is required to analyze attack graphs. In this study, we proposed a Memetic based algorithm for minimization analysis of network attack graphs. Memetic Algorithms (MAs) are a class of stochastic global search heuristics in which Evolutionary algorithms-based approaches are combined with local search methods. Simulated Annealing algorithm is used as local optimizer in presented algorithm. We compared Proposed algorithm with Memetic Particle Swarm Optimization algorithm through different attack graphs. Results proved that the Proposed algorithm is accurate and has better performance.

REFERENCES

Abadi, M. and S. Jalili, 2008. Minimization analysis of network attack graphs using genetic algorithms. IJ Comput. Appl., 15: 263-273.

Alhomidi, M. and M. Reed, 2014. Attack graph-based risk assessment and optimisation approach. Int. J. Network Secur. Appl.

Eiben, A.E. and J.E. Smith, 2003. Introduction to Evolutionary Computing. Springer, New York, USA., ISBN-13: 9783540401841, Pages: 199.

Engelbrecht, A.P., 2005. Fundamentals of Computational Swarm Intelligence. 1st Edn., John Wiley and Sons Publishing Inc., New York, USA., ISBN-13: 9780470091913, Pages: 672.

Krasnogor, N., A. Aragon and J. Pacheco, 2006. Memetic Algorithms. In: Metaheuristic Procedures for Training Neutral Networks, Alba, E. and R. Marti (Eds.). Springer, New York, USA., ISBN-13: 978038733415 8, pp: 225-248.

Lufeng, Z., T. Hong, C. YiMing and Z. JianBo, 2009. Network security evaluation through attack graph generation. World Acad. Sci. Eng. Technol., 54: 412-416.

Moscato, P. and C. Cotta, 2007. Memetic Algorithms. In: Handbook of Approximation Algorithms and Metaheuristics, Gonzalez, T.F. (Ed.). Taylor and Francis, Boca Raton, FL., pp: 27-1-27-12.

Noel, S., S. Jajodia, L. Wang and A. Singhal, 2010. Measuring security risk of networks using attack graphs. Int. J. Next Gener. Comput., 1: 135-147.

Sheyner, O., J. Haines, S. Jha, R. Lippmann and J.M. Wing, 2002. Automated generation and analysis of attack graphs. Proceedings of the IEEE Symposium Security and Privacy, May 22-25, 2011, California, USA, pp: 273-284.