Adaptation of Memetic Algorithm for detecting Polymorphic forms of Script Malware

SushilaAghav
MITcollege Of Engineering ,Pune University

Vishal Ithape
Student,MITcollege Of Engineering,Pune University

Devesh Chaudhari
Student,MITcollege Of Engineering,Pune University

ABSTRACT

A new generation of attacks called as polymorphic attacks - where malware repeatedly mutates to deceive regular malware detection - are continuing to drive the growth in complexity of malware. Polymorphic malwares are using far more sophisticated approaches that may include editing its own source code to avoid signature-based detection. There is increasing necessity to handle this level of unprecedented polymorphism. Especially, scripts have been exploited to widespread polymorphic malwares. In this paper, we propose a modified Hybrid detection model based dependency analysis. Every script malware can be represented by a dependency graph and then the detection can be transformed to the problem finding maximum subgraph isomorphism in that polymorphism still maintains the core of logical structures of malwares. We also present threshold selection and priority level management approaches which can be used to improve detection accuracy and reduce computational cost.

General Terms

Security, Malware, Algorithms, Experimentation,

Keywords

Malware detection, subgraph isomorphism, genetic algorithm, dependency graph

1. INTRODUCTION

Computer Malware is software or program that damages the computer system or interferes with normal operation and does something unwanted to the computer [1,2].Malware propagates overconnected systems. Technically, malware is a broader term that includes viruses, worms, backdoors, exploits, etc. However, the most well-known type of malware is virus; the term which was coined by Fred Cohen in 1983. Computer malware have been a major threat to the computer systems and networks since 1990s. However, the malware sophistication has significantly improved since the early days. Since the beginning, there is a big contest between virus creators and experts and it is becoming more complicating every day, and will continue afterwards. This is a quite general that the major cause for this the new tactics get developed by virus designers every time. Finally, the malware of the current generation use polymorphic techniques to obfuscate their code with every replication. Polymorphic viruses are an example of the malware evolution processes. These are a kind of body-polymorphic, where body of virus itself changes from one instance to another. Metamorphic malware is another variant of malware. Metamorphic malwares use obfuscation techniques to reprogram themselves into a new transformed code. The metamorphic nature of the malware mutates itself while spreading across the network and making signature based detection completely ineffective. In recent times malwares written in script languages like Visual Basic Script, JavaScript, etc., which are distributed in the form of sources have gained popularity and spreading fast. It is easy to make new variants in high level in that the source code itself is a virus. In this paper, we study proposed system to detect polymorphic malwares, address common issues, and discuss solutions.

2. RELATED WORKS

Malware detection depends on the capability of handling obfuscated malware efficiently. Code obfuscation changes malware syntax but not its behavior, which has to be preserved. Both reverse engineering and deobfuscating techniques generally begin with some sort of static program analysis, which can be depicted as an abstraction of program semantics [3].

Most important techniques/methods employed for malware detection [4] are :

Signature-Based Techniques- Most of the antivirus tools are based on detection with signature based techniques. These signatures are created by examining the code of malware binary. Various disassemblers and debuggers are available which help in disassembling the portable executables.Thiscode is analysed and features are extracted. These features are used in constructing the signature of particular malware family.

Behavior-Based Techniques- In this technique main goal is to analyze the behavior of known or unknown malwares. Behavioral parameters include various factors such as source/ destination address of malwares, types of attachments and measurable, statistical features.

The detection process accompanied by signature based and behavior based techniques which further accomplished the whole process with static, dynamic and hybrid analysis.

I. Static Analysis :It is the process of analyzing executable code.In static analysis we extract the low level information from codes which gathered by disassembling the codes with the use of any disassembler tools. Static analysis has the advantage to reveal the code structure of the program under consideration. Static analysis may fails in analyzing unknown malware that uses code obfuscation techniques. [4].

II. Dynamic Analysis : It is also called as behavioral analysis, involves executing the malware and monitoring its behavior, system interaction, and the effects on the host machine. In dynamic analysis, infected files are analyzed in simulated environment like a virtual machine, simulator, sandbox

etc.Dynamic analysis may fail to detect an activity which shows the behavioral changes in codes by different trigger conditions during the course of its execution.
III. Hybrid Analysis :Hybrid analysis includes the combinatorial approach of both dynamic and static analysis. It analysessignature of malware then combine it with the other behavioral,heuristic parameters for enhancement of complete analysis. Due to this approach hybrid analysis overcomes the limitations of both static and dynamic analysis.

Several systems were proposed to exploit a machine learning-based framework. Crowdroidis a machine learning-based framework that recognizes Trojan-like malware. Another IDS that relies on machine learning techniques is Andromaly.MADAMuses a global-monitoring approach that is able to detect malware contained in unknown applications. To overcome hardship in detecting polymorphic viruses in early stages, Several techniques for virus detection were suggested including rule learning, control flow graph, neural networks, and data mining approaches.

3. OBFUSCATION TECHNIQUES

This section introduces the obfuscation techniques commonly used in the polymorphic and metamorphic malware. [4,5]

3.1 Dead-Code Insertion

Dead-code insertion is a simple technique that adds some ineffective instructions to a program to change its appearance, but keep its behavior. However, the signature based antivirus scanners can be defeated by this technique.

3.2. Subroutine Reordering

Subroutine reordering obfuscates an original code by changing the order of its subroutines in a random way. This technique can generate n! Different variants, where n is the number of subroutines. For example, program having 5 subroutines, would lead to 5! = 120 different generations.

3.3 Code Transposition

Code transposition reorders the sequence of the instructions of an original code without having any impact on its behavior.

3.4 Control Replacement

Some control statements perform similar works. For example, a ‘for’loop and a ‘while’loop are interchangeable. The dependency of statements and variables are still kept.

3.5 Instruction Substitution

Instruction substitution evolves an original code by replacing some instructions with other equivalent ones.

4. PROPOSED ARCHITECTURE

Our proposed architecture is a hybrid model consisting of various phases to detect malware using Memetic Algorithm with local search techniques based on existing system [6] proposing few modificationsfor more accurate results.Such as ,Variance as a Stopping Criterion [7], self evolvable threshold ,data categorization and priority levels.Priority levels can be used in combination with importance of functions and dependence of nodes for discarding wrongful result.There are various phases in the proposed model. In the **phase 1,** Malware or suspicious file is parsed and transformed to a code with semantic meaning. Each line of the original code is divided into a number of tokens. While parsing necessary information is gathered and data is categorized by using matching rules, patterns to extract variable use-definition table as well function calls. Data categorization would reflect type of operation like system call, assignment operation, arithmetic operation, object creation etc. Semantic code can be stored using symbol table, dependency table and later transformed into dependency graph.
Phase 2, includes the two important functions, Dependency Graph[8] Generation,priority level management. The system extracts a dependency graph from the Dependency table and priorities are handled at same time using information added in phase1.
Phase 3, the system extracts a dependency graph and
Conducts graph reduction to diminish the size of graph. The size of a dependency graph may be reduced. Redundant, dead code can be removed with certain care taken to preserve underlying structure and dependencies. With exception of vertices with highest priorities any other vertex that satisfies one of the following four conditions can be eliminated:
• A vertex with only one outgoing edge without any incoming edge represents declaration of a variable which can be ignored.
• A vertex with no edge is dead code/variable and can be removed.
Phase 4, is very important in such a way that it is used to determine initial threshold value, which may be improved with each generation later in GA. The system compares the dependency graph with the target graph by using fitness function to compute difference value.
Phase 5, in this phase, using a hybrid GA, the system, attempts to find polymorphic variant of malware. Genetic operators[8] are used to form generations of solutions and best of solution is chosen with help of exhaustive search until stopping criterion is met.
Phase 6, in this phase final comparison reports are generated. It is connected to obtain final analysis and results.

4.1. Priority Levels

In **phase2** on occurrence of System functions priority levels are raised, Such as VBScript uses scrrun.dll for File system management, file modification, and streaming text operations etc. In such case priority level would be set to maximum. In **phase3** after reduced graph is generated priority levels are set according to total incoming and outgoing edges per node. Consider 4 priority levels, 1 being maximum priority and 4^{th} lowest priority. Increment Factor= (Maximum no of Total edges – Lowest no of Total edges) / (Priority Levels+1) E.g. Consider, Max total edges 12 and Min total edges 1.So partitions would be as follows with range 2 and last partition of remaining set of edges like, 1-3, 4-6, 7-9, 10-11 .We can use priority levels to determine importance of node while comparing graphs as well comparison of associated data with each node can be extracted and can be compared with data categorization completed in phase 1.

4.2. Encoding, fitness function and selection strategy

To find polymorphic variant of malware using dependency graphs, problem can be broken down to subgraphisomorphism[9]. Subgraph isomorphism is an NP-hard problem so GA is suitable solution to find combinations. In memetic algorithm local search techniques are used. Let's discuss over some of important GA terms and their adaptation for solving graph isomorphism problem. It is natural to select problem permutation encoding as encoding scheme for the solution candidates of the graph matching. Since this is a common scheme, several well studied operators are available. The fitness of an individual solution is determined by the .An individual is fitter if it has a smaller value. Tournament selection, different from, fitness proportionate or rank based selection strategies, does not require either normalization or ordering of fitness values.

4.2.1. Replacement, mutation and stopping criteria

Crossover operators for permutation encoding problems can be divided into three families, depending on which kind of information is preserved. Order based crossover (e.g.OX [10]) preserves relative position, while position-based crossover (e.g., CX [10]) focuses on the absolute position of the alleles. A popular hybrid approach is partially mapped crossover (PMX [10]), which exploits simultaneously the ordering and the absolute positions. Cycle crossover and PMX may be used. Strict position based crossover (PBX [10]) provides viable option.

4.2.1.1. Replacement

We replace the worst members of the population with the new 20% offspring.

4.2.1.2. Stopping criterion

GA stops when number of generations Exceeds pre-defined value or if no further improvement in the best fitness value for consequent generations is observed for a fixed number of iterations. .We may use Variance as a Stopping Criterion for Genetic Algorithms [7]. Variance should be greater than pre-defined value. Which is negligible positive value (ε) e.g. 0.005.It represents least deviation from average fitness value of best individuals.

4.2.1.3. Mutation or Local optimization

For the mutation simply swap operator could have been used. Which simply exchanges the loci of two alleles? Its effect would be to maintain the population diversity during evolution. However, for Memetic algorithm, in contrast to the standard genetic algorithm, the use of mutation does not improve either Solution quality or convergence speed. Moreover, the local search step works like an intelligent swap mutation, and can thus replace the ordinary mutation completely. The local search strategy of the Memetic algorithm consists in replacing each offspring by the dominating gene of its neighborhood.

5. FUTURE WORK

In the future, we are planning to apply the algorithm with self evolving threshold within generations while performing GA. We also think that better local search techniques and further parameter optimization as well including priorities ingraph matching phase with more self evolutionary methodsin general might still improve the algorithm'sperformance.

6. CONCLUSIONS

In this paper, we proposed a Modified hybrid model consisting of various phases to detect malware using Memetic Algorithm. This detection mechanism is based on the dependency analysis by using Genetic Algorithm with Local Search Techniques. To detect unknown polymorphic malwares with priority levels and adaptable threshold value in GA seems viable option. Malware detection problem is simply broken down to maximum subgraphisomorphism problem and we use a Memetic Algorithm to find feasible solution. Especially, graph reduction with priority levels and the local search approaches also prove to be important parts of approach followed.

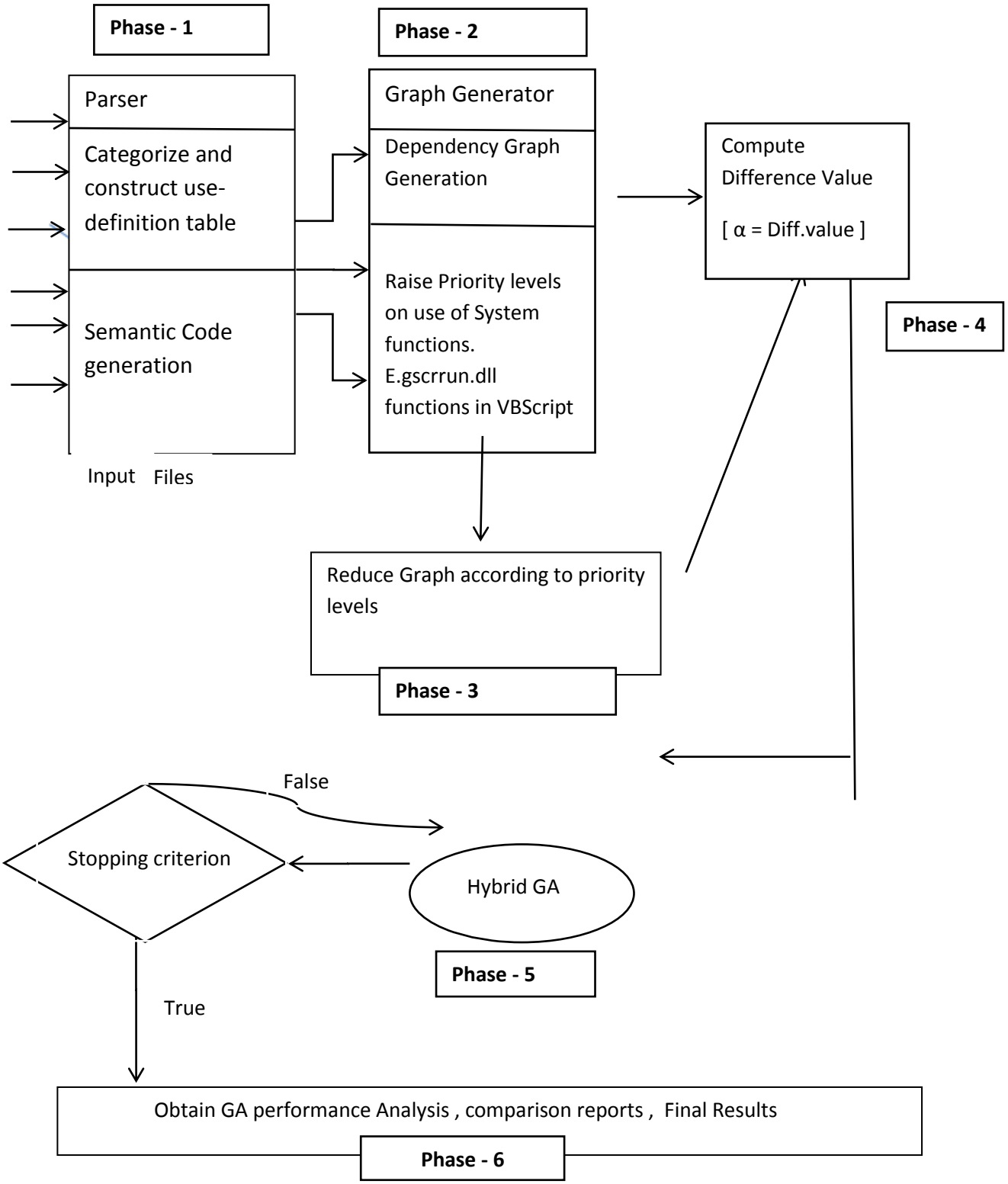

Fig 1 : Overview of proposed architecture

Initial Threshold value [α] is set using Difference Value.

7. REFERENCES

[1] A Survey on Techniques in Detection and Analyzing Malware Executables , IJARCSSE Volume 3, Issue 4, April 2013

[2] J. Aycock. *Computer Viruses and Malware*. Springer, 2006.7

[3] Mila DallaPreda: Code Obfuscation and Malware Detection by Abstract Interpretation Universit`adegliStudi di Verona, Dipartimento di Informatica, TD-02-07, 2007.

[4] Ilsun You and KangbinYim: Malware Obfuscation Techniques: A Brief Survey, International Conference on Broadband, Wireless Computing, Communication and Applications, 2010.

[5] AriniBalakrishnan, Chloe Schulze "Code Obfuscation Literature Survey"

[6] Keehyung Kim, Byung-Ro Moon "Malware Detection based on Dependency Graph using Hybrid Genetic algorithm"

[7] DinabandhuBhandari, C. A. Murthy, Sankar K. Pal "Variance As A Stopping Criterion For Genetic Algorithms

With Elitist Model"

[8] J. Ferrante, K. J. Ottenstein, and J. D. Warren "The program dependence graph and its use in optimization,"

[9] Thomas B¨arecke and MarcinDetyniecki "Combining Exhaustive and Approximate Methods for Improved Sub-Graph Matching"

[10]AbdounOtman, AbouchbakaJaafar "A Comparative Study of Adaptive Crossover Operators for Genetic Algorithms to Resolve the Traveling Salesman Problem" IJARCSSE,volume 3 , issue 4 , 2013

[11] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley

Longman Publishing Co., Inc., 1989.

[12] Jonathan A.P. Marpaung, MangalSain and Hoon-Jae Lee: Survey on malware evasion techniques: state of the art and challenges, International Conference of Advanced Communication Technology, pp 19-22, 2012.

[13] RizwanRehmani , G.C. Hazarika and GunadeepChetia : Malware Threats and Mitigation Strategies: A Survey, Journalof Theoretical and Applied Information Technology, Vol. 29 No.2, 2011.

[14] I. Oliver, D. Smith, and J. Holland, "A study of permutation crossover operators on the traveling salesman problem," in *Proc. of the 2nd Int.Conf. on Genetic Algorithms*, Mahwah, NJ, USA, 1987, pp. 224–230.6

[15] JacoboToran´ "on the hardness of graph isomorphism"

[16] Thomas B¨arecke and MarcinDetyniecki "Memetic Algorithms for Inexact Graph Matching"