Advances in Computer Science Research, (ACSR), volume 52
2016 International Conference on Computer Engineering and Information Systems (CEIS-16)

An Extensible Anti-Virus Framework Design Scheme

Ya-Juan Yuan, Ling Ding
Cang Zhou medical college
E-mail: hbdlyyj@163.com, 273366693@qq.com

***Abstract*-Anti-Virus engine depends too much upon the signature database and each detection module couples tightly. To solve this problem, an extensible framework is presented. The function module and data module are decoupled by using the intelligent signature and decomposing composite modules. Inversion control is used to change the control coupling into the interface coupling in the modules of Anti-Virus engine. The extensible framework not only solves the disadvantages of the Anti-Virus software, but also applies to the design of embedded security device. A prototype is implemented to verify the validity of the proposed scheme and key algorithm. The experimental results show that the novel Anti-Virus engine can detect the virus more effectively than traditional models.**

Keywords-anti-virus engine; behavior signature; virus variants; unpack

I. INTRODUCTION

In recent years, with the spread of information crime presents the two development trends, the gray industry chain of computer virus using virus as the profitability tools is increasingly mature and the virus generating groups using the Internet operation mode leads to the virus proliferation and faster spread. According to the report of the early warning detection system of the computer virus, we seized nearly 1.1 million kinds of computer virus in 2008, up to 201.9% from 2007. The main virus among them is the Trojan virus (78%), the backdoor virus (10%), and advertising virus 4% [1]. The data shows that a huge number of computer virus is growing rapidly, and the main threats come from Trojans and backdoor virus. The other is a computer virus variant speed, for example, in 2009; the "Meg." virus produces 300 variants in less than two months [2]. What leads to the increase of the virus variant speed is the direct cause of the popularity of packers, which is free to kill fuzzy tools code. According to figures from the rising company network, there is thousands of packer application software and ordinary computer users using packers or other code fuzzy tools can easily create pandemic virus variants. In short, the current quantity and diversity of computer virus is increase, the spread and variant speed has increased rapidly, and the anti-virus industry has encountered the huge challenge.

In this paper, we study the antivirus engine extensible framework, which can realize the detection module increase or decrease based on a new framework. Prototype system is designed on the basis of the open source software AnSav [9] [11]. Section 2 introduces the related concepts of the research question. Section 3 presents the antivirus engine frame structure, the key algorithm, etc. Section 4 is the simulation part. Finally, section 5 summarizes the work.

II. THE DEFINITION OF THE RELATED CONCEPTS

Definition 1 (Intelligent feature codes) intelligent signature consists of two parts, < behavior code, and virus signature code> [2]. Code of behavior is a sequence of virus behavior characteristics, to facilitate the formalization expressed of algorithm as Sig = < Sig1, Sig2... Sign>. One Sigi represents some kind of behavior, such as file operation, control process, etc.

Definition 2 (Atoms Atom_Activity): Atomic activity is the most basic activity unit of antivirus engine, it is not divided. Each atom activities including d ∈ D and input and output part of activity. Input part consists of event trigger conditions, including the input parameter Xvi, prerequisite Xpi and required resources Xri. Output is that executes activities result data Xvo and follow-up conditions Xpo. Atomic activity is divided into three broad categories on types: pretreatment activity (marked P), scanning (marked S) and remove activities (marked C).

Definition 3 (Inspection activities): Inspection activities are set composed of binary < Rules, Atom_Activities > collection. The nature of the testing activity is atomic activity sequences with a consistent target, Rules is logic of reasoning for construction atomic sequence, and Atom_Activities represent atomic activity sequences.

Definition 4 (Rules): Rules are described detection logic reasoning process, can be represented as . is the precondition for rules, but is the form parameter of rules application. Postposition state is the detection logic generated by rule.

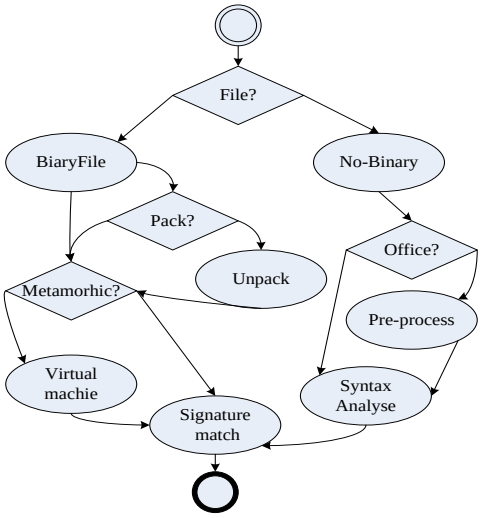

Figure 1. File type testing activity diagram.

In order to explain the above concept, figure 1 describes antivirus engine file detection module, a visible detection activity is a set of defined 3 atomic activities, including diamond denotes judgment rules, and oval represents atomic testing activities. Figure 1 the first diamond signifies rules of determine file type, the execution direction of the testing activities is decided by file type. In addition, testing activities contains many kinds of atoms activity. Unpack activities, for example, Office files syntax analysis is belongs to the typical pretreatment activities; Virtual machine detection, signature matching are belong to scanning activity.

Definition 5 (Detection logic): Detection logic is the finite set form such as L = {P1... S1, Pm, C1...}. Detection logic essence is an ordered set of atomic activity, at the same time, the following properties: the concurrency, there are ordered subset L1... Li (i > 0), each detection logic sequence execution time for T1... Ti, and meet the conditions, the execution time has no obvious constraints. Dependence, both the subsequence of the execution time of the execution shape the constraints of the requirement such as Ti > Tj.

III. EXTENSIBLE FRAMEWORK DESIGN SCHEME

Extensible framework design by the Inversion of Control pattern inspired IC (Inversion of Control). Inversion of control pattern is a new type of software design patterns, belongs to the important application in architectural technology mode, can solve the dependencies between components. Its essence is inverse the dependencies between module, by a special container to specific configuration, make no code layer correlation between module [12]. All testing activities of antivirus engine has certain dependencies, this dependency is detection logic. Anti-virus extensible framework's goal is to reduce each testing activities control coupling as far as possible, kept interface coupling relationship between testing activities, delete, add a module does not need to change the entire framework.

A. The Structure of the Extensible Framework

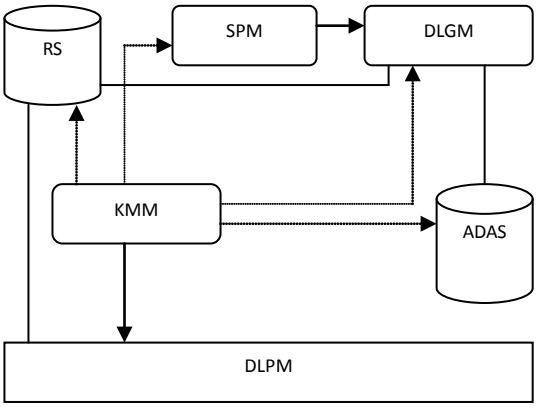

Figure 2. Extensible framework architecture diagram.

KMM (Kernel Management Module). KMM implements schedule of each Module, responsible for starting the extensible framework, operation the update task of atom activities set. As shown in figure 2, dotted line represents KMM send control information to other modules. For ordinary computer virus KMM directly control DLPM work, performing routine virus detection logic; For a specific behavior of the virus, KMM start SPM parsing behavior characteristics code, and then entered the stage of dynamic adjustment engine, adjusted to detect specific virus. The following focuses on engine for a specific virus detection process, feature code parsing Module SPM (Signature Parse Module) is responsible for parsing intelligent feature codes, atomic activity concentration in the use of signature code information retrieval is suitable for testing the atom of a particular virus activity. DLGM (Detection Logic Generate Module).

B. Design by Rules

First we analyze the specific behavior of computer viruses, with the results of the analysis as a priori knowledge, reasoning knowledge base is established. Spatial reasoning knowledge base is essentially rules space, defining the conjunction, disjunction, not three operators, using assembly language of macro operators AND, OR and jump statements to achieve the three operations. Using a conditional statement said preconditions of the rules $\varphi(X_1, X_2, ..., X_k)$, form parameters of the preconditions of as the actual parameters receiving behavior signature; the output of the rules is a poset of testing activities. For example, a new special virus UPX packer's algorithm notes for UPXn, fuzzy entrance OEP behavior. Because to correctly detect the virus, you must first add new unpack module completed, reprocessing OEP ability after feature matching, so the inference rules are:

R: IF Sigupxn AND Sigoep then < < Pupxn, Poep >, S >

Sigupxn, Sigoep for actual rules parameters, AND is the logic operation of rule conditions $\varphi(X_1, X_2)$, < < Pupxn, Poep >, S > is a poset of testing activities. Partial order

relation < Pupxn , Poep > denote hulling activity Pupxn first, later in the treatment of the Poep; , partial order relation < < Pupxn, Poep > S > denotes that after the < Pupxn, Poep > executes, the feature matching S is performed.

C. Smart Code Parsing Algorithm

In order to solve the incomplete problem of libraries of testing activities BASet, first perform SPM analysis of the behavior characteristics of specific virus, according to the information of behaviour from extension atomic inspection activities in the library EASet search activities. EASet = {T1, T2,..., Tm} is to extend the atomic activities, contains a variety of specific testing activities, with a tree structure. In a particular test activities aj (i = 1... m) as the foundation to create tree, ai is obtained by first order through calendar calculation method of sequence of activities to perform aj poset. Such as first sequence traversal tree Tj get sequence < ao, ax, ay, aj >, instructions to perform aj must order three testing activities ao, ax, ay.

Input: Sig=<Sig1, Sig2… Sign>
Output: Testing activities aj(j=1..m) compose set SET
(1) Begin
(2) Initialize (BASet);
(3) Initialize (EASet);
(4) For each Sigi in Sig
(5) IF match (Sigi,aj)==True AND aj not in BASet
(6) For each Ti in EASet
(7) IF FOUND (aj, Ti) ==True
(8) aseti = Preorder(aj, Ti);
(9) ENDIF
(10) ENDFor
(11) SETa←aseti;
(12) ENDFor
(13) END.

IV. CONCLUSION

In this paper, we solve the antivirus engine control coupling problems between the components using the method of composite testing activities which are decomposed into atomic engine implements and the dynamic adjustment testing process, it also raise the new virus and variant of the virus detection ability. The antivirus engine design scheme can be applied to the embedded security devices, such as embedded network gateway and anti-virus wall with low maintenance costs. The effectiveness of the design scheme is verified by a prototype system. Intelligent feature codes can direct the work of engine dynamic adjustment and improve the accuracy of the anti-virus engine. Because the engine is optimized for a specific virus treatment process, it reduces the number of large computing time testing activities and obtains good performance. We plan to go on with the improvement of the prototype system, the optimal detection logic generation algorithm, the increase of the accuracy of the detection logic. On the other hand, we will improve the detection logic adjusting module and reduce the engine adjustment process of time delay.

REFERENCES

[1] G. Szappanos.2002. "Are There Any Polymorphic Macro Viruses at All? (... and What to Do with Them)," in Proceedings of the 12th International Virus Bulletin Conference.

[2] P. Ször and P. Ferrie.2001. "Hunting for Metamorphic," in Proceedings of the 11th International Virus Bulletin Conference.

[3] G. Wroblewski.2002. General Method of Program Code Obfuscation, Institute of Engineering Cybernetics, Wroclaw University of Technology, PhD. Thesis.

[4] V. Bontchev.2002. "Macro and Script Virus Polymorphism," in Proceedings of the 12th International Virus Bulletin Conference.

[5] M. Christodrescu and S. Jha. 2003. "Static Analysis of Executables to Detect Malicious Patterns," in The 12th USENIX Security Symposium (Security '03), Washington DC, USA.

[6] Brand, M. 2007. Forensic Analysis Avoidance Techniques of Malware. Paper presented at the 5th Australian Digital Forensics Conference, Edith Cowan University, Mount Lawley Campus, Western Australia.

[7] Bridges, L. 2008. The changing face of malware. Network Security, 2008(1), 17-20.

[8] Regshot. 2008. Regshot. Retrieved August 25, 2008, from http://sourceforge.net/projects/regshot