Indian Journal of Science and Technology, Vol 10(20), DOI: 10.17485/ijst/2017/v10i20/110608, May 2017
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

A Pragmatic Way of Logic Bomb Attack Detection Methodology

Bobby Sharma*

School of Technology, Assam Don Bosco University, Guwahati – 781017, Assam, India;
Bobby.sharma@dbuniversity.ac.in

Abstract

Objective: This study proposed and developed a frame work to detect the insider and outsider logic bomb attack in a system. **Method/Analysis:** On fulfillment of certain conditions, ongoing system may suffer from various irregularities including system integrity failure, auto file deletion, auto updating, buffer overflow, memory synchronization failure etc. It is difficult to realize the existence of Logic Bomb attack. Malicious codes are hidden inside the main file or embedded in hardware. Even such codes can be injected remotely. Evidence generated from the studies show that detection and diffusion of Logic Bomb attack in advance is difficult. Even testing of Logic Bomb attack that is embedded in hardware need well equipped testing devices. However, systematic approach of observation and analysis help to detect logic bomb attack. In the proposed methodology, a framework has been generated in which it incorporates various factors of irregularities from system based observations and data extracted from firewall. **Findings:** As Logic Bomb attack does not have any stereotype approach. Thus it creates more complexity. It needs minute observations. Proposed method has been implemented for both inside and outside logic bomb attack and results are compared. **Novelty/Improvement**: Depending on types of consequence as well as observations, proposed methodology can be extended further.

Keywords: Auto File Deletion, Buffer Overflow, Firewall, Logic Bomb, Remotely Injected, System Integrity Failure, Stereotype, System Based

1. Introduction

Logic bomb attack is nothing but a program segment which is included in the main program for triggering malicious action under certain condition[1-5]. These are basically hided inside or may be out-of-spec code to go off after meeting certain condition[6]. A programmer known as Tony Xiaotong was charged for keeping logic bomb in the application during his employment. He had embedded malicious code in the application which was supposed to blast after specified time. Luckily, it was caught and diffused with several efforts before time. Similarly, code had been changed in the server which was set to blast on a birth day. But due to error in the code, it was not blast. Attacker tried once again but it was detected before time[7]. Like this, several other examples are available.

It may be embed within standalone programs or more dangerously they may work like a part of a worm or viruses. These may also be called as malware[8]. When they work as a part of virus, it generates multiple copies of it and spread in such a way that it becomes uncontrollable which is known as flooding[9]. It may be Time Bomb which is a subclass of logic bomb, explode or activate at certain time as specified by the programmer. As for example "Friday the 13th" this activates on specified day and date and corrupts the files. Another one which activates on 6th of March and corrupts the hard disk. Apart from these another very dangerous logic bomb attack which is also known as hardware Trojan which may be created during the time of fabrication of the hardware can explode as time bomb if the Trojan works as sequential state machine[2]. Network based monitoring as well as multi level security policy can be implemented for proper monitoring and detecting malicious code[10]. Hardware Trojan detection relies on IC testing or system level testing which generates limitation for logic bomb detection process[11].

*Author for correspondence

Logic bomb may also be of type Arbitrary Code in which it may insect code from outside to corrupt the system after meeting certain specified conditions[1]. Logic bomb in Android application is another problem faced by the user. Though, Google Bouncer and Manual application review or similar kind of automated approaches have been adopted still it becomes difficult for the user to understand all kind of malicious piece of code and diffuse them from current applications[12].

It is generally activated in the presence or absence of some files or on particular date & time or for particular user. It also damages system or modifies the system or delete files[13,14].

Authors explained that it is not possible to provide an unconditional guarantee of invulnerability to intrusion[3]. Out of several reasons, logic bomb is one of the reasons for this. Even, logic bomb never publishes itself but after meeting the given criteria they explode[14,15]. It has certain flow which makes it happen only for certain applications not to replicate towards other application[16]. Logic bomb can also be defined as a type of time bomb[17]. Some insider attackers used logic bomb to attack specific systems like social engineering, compromising account, unauthorized access of account, editing of log files etc.[18]. Attacker may deceive the user easily by restructuring the file format. A pdf files can be recognized as one of such files. Of course malicious pdf files can detect by analyzing their file format[19]. Though the logic bomb can be implemented in variety of fields still some of classic field of application of logic bomb are to get payment or to make free software trial or to halt some ongoing activity etc[20]. History of past scans is generally not stored in logic bomb attack[4]. That makes it more difficult to detect. In[21], author mentioned about the damage caused by Sybil Logic Bomb which is almost as severe as the Great Financial Crisis of 2007-2012. It is mentioned that cyber attacker uses the technique of logic bomb for creation of intrusion.

In[12], authors tried to detect logic bomb attack automatically. The main logic behind this method is that Logic bomb is basically occurred only after specific circumstances. Hence detection process carefully observes the different checks being implemented in different points where there is a chance of occurring the logic bomb attack. They give less importance to the behavior itself. "Trigger analysis" which is a static analysis system, combines traditional program analysis along with novel elements used for automatic identification and detection of triggers. After that to identify the interesting check, they are using predicate reconstruction, path predicate minimization and predicate classification.

In[19], author mentioned that PDF files have the capability to bear lots of malicious code along with them. Here, authors proposed a novel evasion technique called reverse mimicry which uses real samples to detect such misbehavior. Here, authors believed that for a PDF that carried malicious code, a structural detection model can be evaded by reverse mimicry attack. It has been experimented and validated that malicious PDF files can be analyzed and detect attacks.

In[22], it is mentioned that proper characterization as well as management of hardware, software can help in preventing and detecting the logic bomb attack. Logic bomb attacker may corrupt the system remotely by autonomous agent. Though it is difficult to detect such attack yet organization can create a trusted baseline for all machines in the network, maintain a secure location which will be observed periodically for new footage. Any updated or modified data can be compared and analyzed to detect adversary. Author also mentioned about technical controls with multiple level of authorization.

2. Defense against Logic Bomb Attack

System can be protected from Logic Bomb Attack by taking several defensive measure like by separating the process of testing and coding, implementation of code, people must be well authorized for the updating, implementation or execution of code, each user should have separate authentication code, be conscious of any spam files, periodic execution of program, keep track of any new updating and its after effect etc. Regular monitoring of file system integrity failure as well as monitoring of log file can also prevent logic bomb attack. Specific attention must be given for auto alteration or installation of software. This may lead to file system integrity failure for a short period. In such situation, there may a suspected logic bomb attack. Any spam or malicious data must be protected by anti spam or firewall kind of software. Protecting the system at end user level is most important. Network/system administration must take care of different areas including regular backups, scuffling of duty, strong end point protection etc.

Manual checking and monitoring of the system may sometimes be quite difficult. Hence in the following methodology, auto monitoring and detection system for logic bomb attack is proposed.

3. Proposed Detection Process

In general, Logic Bomb attacks are insider attack in which privileged user may manipulate the code or alter the system in such a way that as per criteria spontaneously activates and attack the system. In such situation, memory latency or size may increase. Frequent buffer overflow or system failure may occur. B

It is not necessary to categorized Logic Bomb as Insider Attack. Any user who may get the privileged access to system may impose Logic Bomb attack to the system from distant. Thus leveraging the vulnerability to the system. In such situation "Firewall" may play a vital role to understand which packet to be filtered. Even then, attacker may manipulate the system administration to gain access to privileged function. Hence in the proposed detection process, it emphasizes the system level detection as well as abnormality recorded by Firewall.

Based on the misbehavior observed in the system, the proposed methodology initially maintains the record as suspected Logic Bomb Attack. If it happens for more than a threshold number of time then it identifies it as "Logic Bomb Attack.

Algorithm to detect Logic Bomb Attack

Input: $m_l, m_{lt}, m_s, m_{st}, f_b, f_{bt}, f_d, f_{dth}, s_f, s_{ft}, pkt, pkt_{th}, sus, sus_{th}, i\_sus, i\_sus_{th}$

Output: Detection of Logic Bomb Attack

1. Begin
2. scan *logfile*
3. If the *logfile* contains "*new activation of program OR periodic activation of program*" then
4. evaluate m_l {* m_l is the temporary latency *}
5. evaluate m_s {* m_s is the occupied memory size *}
6. evaluate f_b {* f_b is the frequency of buffer overflow *}
7. evaluate f_d {* f_d is the file deletion record/file system integrity disable *}
8.
9. If ($m_l > m_{lt}$) then
10. find m_s {* m_s occupied memory size *}
11. If ($m_s > m_{st}$) then {* m_{st} occupied memory size *}
12. find f_b {* f_b frequency of buffer overflow *}
13. find s_f {* s_f frequency of system failure *}
14. end if
15. end if
16. find *pkt* from firewall ; {* *pkt* is the number of packets dropped by firewall *}
17. If ($f_b > f_{bt}$ OR $s_f > s_{ft}$ *OR* $pkt > pkt_{th}$ *OR* $f_d > f_{dth}$) then {* f_{bt} is the threshold frequency of buffer overflow, s_{ft} is the threshold frequency of system failure, pkt_{th} is the threshold packets dropped by firewall, f_{dth} is the threshold file deletion/ file system integrity disable *}
18. find *sus* ; {* *sus* is the number of time of occurrence of "Suspected Logic Bomb Attack *}
19. *sus*++;
20. If ($sus > sus_{th}$) then {* sus_{th} is the threshold number of time of occurrence of suspected logic bomb attack *}
21. Print "Logic Bomb Attack" ;
22. Else
23. print "Suspected Logic Bomb Attack"
24. Update sus by sus++;
25. end if
26. end if
27. end if
28. end if

4. Implementation, Results and Analysis

For implementation, randomly both inside and outside Logic Bomb Attack has been imposed to a system. At

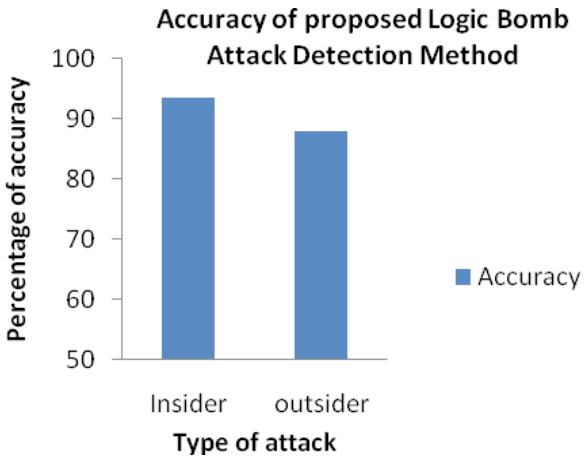

Figure 1. Comparison of accuracy of detection of Insider and Outsider Logic Bomb Attack.

the same time proposed methods for detecting Logic Bomb attack is also implemented to the system. As the attack took place, the proposed system tried to detect the attack.

It is observed that accuracy of the detection is more in case of insider attack in comparison to outsider attack which is reflected in Figure 1.

5. Conclusion

Logic bomb attack is very difficult to detect in advance. It may paralyse the ongoing system in multiple ways. However, systematic way of approach as well as structural analysis can help determining logic bomb attack. Occurrence of logic bomb attack has been observed from various factors including auto activation of new program or periodic occurrence of program, observation in terms of latency, memory occupancy, and buffer over flow as well as frequency of system failure etc. If some abnormalities occur then initially it is kept under suspected logic bomb attack. If the number of suspected logic bomb attack crosses the threshold number of times them it determines as confirmed logic bomb attack. The proposed method extracts various data from firewall to determine file system integrity failure along with other parameters. Implementation has been done for both inside and outside attack. Results are compared and plotted in the graph in Figure 1. Efficiency of the proposed methodology is found more in terms of insider logic bomb attack. By this, it leaves future work to improve the efficiency of the proposed method in terms of outsider attack.

6. References

1. Kabay M. Logic Bombs. Dangerous Cargo. 2017. Available from: Crossref
2. Chakraborty R, Narasimhan S, Bhunia S. Hardware Trojan: Threats and emerging solutions. IEEE High Level Design Validation and Test Workshop (HLDVT); 2009. p.1-6.
3. Chalurkar S, Khochare N, Meshram B. A tool to detect and prevent malware attacks: A survey. International Journal of Computer Networks and Wireless Communications. 2012; 2(1):1-7.
4. Rane V, Rane C, Shelar M, Pinjarkar V. Website security tool. International Research Journal of Engineering and Technology (IRJET). 2016; 3(3):1-6.
5. Robillard N. Diffusing a Logic Bomb. GIAC Security Essentials Certification (GSEC). 2004; 1.4b(1):1-100.
6. William TY, Memory A, Henry GG, Senator TE. Detecting unknown insider threat scenarios. IEEE Security and Privacy Workshops; 2014. p. 1-7.
7. Bist AS. Classification and identification of Malicious codes. IJCSE. 2012; 3(2):1-6.
8. Mukkamala S, Sung A, Abraham A. Cyber-security challenges. Designing Efficient Intrusion Detection Systems and Anti-Virus Tools. Taylor and Francis Group, LLC.; 2006.
9. Bist A. Classification and identification of Malicious codes. IJCSE. 2012; 3(2):202-211.
10. Nguyen T, Gondree M, Khosalim J, Shifflett D, Levin T, Irvine C. An approach for cross-domain intrusion detection. 7th International Conference on Information Warfare and Security; 2012. p. 1-11.
11. Wu TF, Ganesan K, Hu YA. TPAD. Hardware Trojan Prevention and Detection for Trusted Integrated Circuits. IEEE Transactions on CAD (TCAD-2015-0006). 2015; 35(4):521-34. Crossref
12. Fratantonio Y, Bianchi A. Robertson W, Kirda E, Kruegel C, Vigna G. Trigger scope: Towards detecting logic bombs in android applications. IEEE Security and Privacy (SP); 2016. p. 1-20.
13. Stallings W. Cryptography and Network Security. 5th ed. Prentice; 2011.
14. Singh AP, Handa SS. Malware detection using data mining techniques. Journal of Advanced Research in Computer and Communication Engineering. 2015; 4(5):1-6.
15. Fortinet, Inc. Advanced Threats, Advanced Solutions: Integrating a Sandbox into Your Infrastructure. 2016. Available from: Crossref
16. Khari M, Bajaj C. Detecting computer viruses. IJARCET. 2014; 3(7):2357-64.
17. Pradeep K, Kumar M. Intrusion detection system for malicious traffic by using PSO-GA algorithm. International Journal of Computer Science Engineering and Technology. 2013; 3(6):1-3.
18. Cappelli DM, Caron T, Trzeciak RF, Moore AP. Spotlight On. Programming techniques used as an insider attack tool. Available from: www.cylab.cmu.edu
19. Maiorca D, Corona I, Giacinto G. Looking at the bag is not enough to find the bomb: An evasion of structural methods for malicious PDF files detection. Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security (ASIA CCS'13); 2013. p. 119-30.
20. Zeidanloo HR, Tabatabaei SF, Amoli PV, Tajpour A. All about Malwares (Malicious Codes). Proceedings of the International Conference on Security and Management (SAM); 2010. p. 1-8.
21. Ruffle, Bowman G, Caccioli F, Coburn AW, Kelly S, Leslie B, Ralph D. Stress test scenario: Sybil Logic Bomb Cyber

Catastrophe. Cambridge Risk Framework series, Centre for Risk Studies, University of Cambridge. Cambridge Centre for Risk Studies, University of Cambridge Judge Business School. 2014. p. 1-45.

22. Keeney M, Kowalski E, Cappelli D, Moore A, Shimeall T, Rogers S. Insider Threat Study: Computer System Sabotage in Critical Infrastructure Sectors. U.S. Secret Service and CERT Coordination Center/SEI insider Threat Study. Computer System Sabotage in Critical Infrastructure Sectors. 2005. p. 1-45.