ISSN: 2278 – 7798
International Journal of Science, Engineering and Technology Research (IJSETR)
Volume 2, Issue 2, February 2013

Analyzing the Impact of Coupling Intensity on Software's Future Attack Surface in Object Oriented Design

Pankaj Pandey, Prof. Niket Bhargava

Department of Computer Science & Engineering,
Bansal Institute of Science & Technology
Bhopal(M.P.)
pankaj82gh@gmail.com
niket.bhargava@rediffmail.com

***Abstract*— Software security failures are common and a long standing challenge to the research community. We can conceptualize the vulnerability of an application through its attack surface size. A system's attack surface is an indicator of the system's security. Unfortunately predicting software's future attack surface size during design phase in earlier stage of software development life cycle (SDLC) is largely missing.**

Our objective is to analyze the impact of coupling intensity on software's attack surface in an object oriented design environment. Here we investigate the statistical relationship between system's attack surface with coupling intensity of an object oriented system.
In this research paper, we investigate whether coupling intensity as a measure of artefact of object oriented software can be utilized as early indicators of system's future attack surface size. For an experimental setting, nine open-source java-based projects were analyzed.

Our experimental results indicate that architectural information from the non-security realm such as coupling intensity is useful in system's attack surface size prediction.

***Keywords*—** Attack surface, Coupling Intensity, Software Engineering, Coupling, Object Oriented

I. Introduction

Motivation: There is growing dependency valuable assets on computer in modern world. Think of the current prevalence of human live dependency on online banking, ecommerce, online airline reservation etc. We see that modern world demands secure software as a core requirement for human lives. Unfortunately, software security failures are increasing exponentially [1].Vulnerability and other faults in a software under construction manifest at code level though they have injected during design phase of software under construction. Preventing Vulnerability during design phase is a great idea to design secure software systems.

Coupling Intensity measure the level of collaboration (coupling) between the operations in an object oriented designing of a software, i.e., how many other operations are called on average from each operation. Very high values suggest that there is excessive coupling among operations, i.e., a sign that the calling operation does not"talk" with the right "counterpart".

Coupling Intensity in a system can be mathematically expressed in the form of design metrics as
Coupling Intensity=CALLS/NOM;
Where CALLS is a software design metrics which counts the total number of distinct operation calls (invocations) in the project, by summing the number of operations called by all the user-defined operations. If an operation foo() is called three times by a method f1() it will be counted only once. If it is called by methods f1(), f2() and f3(), three calls will be counted for this metric.

& NOM Number of Operations, i.e., the total number of user defined operations within the system, including both methods and global functions (in programming languages that allow such constructs).
Hence, Coupling Intensity represents system coupling as it is a computed proportion from two coupling design metrics: CALLS & NOM.

.
On the other side vulnerability of an application can be conceptualized through its attack surface size. A system's attack surface is "the set of ways in which an adversary can enter the system and potentially cause damage" [3]. So, the smaller an attack surface, the safer from harm and more secure the system. The channels that facilitate an attack include system entry and exit points (method calls), input strings, and data items (files). These channels are all considered attack resources that affect the size of an attack surface.

An attacker uses a system's methods, channels, and data items present in the system's environment to attack the system.

Manadhata and Wing used the entry point and exit point framework to identify the resources that are part of a system's attack surface[3].
Historically, attack surface was an informal concept loosely tied to the amount of system functionality that an attacker could access or influence from outside the system [4]. Manadhata and Wing [5] quantify attack surface in terms of the resources used by a system to interact with its external environment. The method they propose for measuring attack surface serves two purposes. First, the attack surface metric numerically characterizes the touchpoints that a system has with its external environment. Second, the metric serves as a prediction system to enable the assessment of security-related risks.

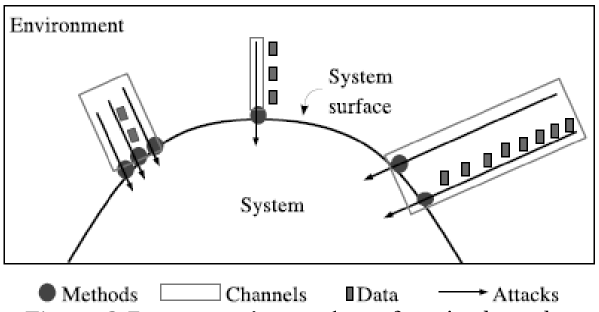

Figure 2.7 a system’s attack surface is the subset of the system’s resources (methods, channels,and data) .

II. Background and Related Work

Attack surface has been quantified through much iteration by research community, scholars and security professionals. Attack vectors [6] was One of the earlier methods used to quantify attack surface. An attack vector is a feature such as a file with weak permissions, an unpatched loophole in software execution, a possibility of buffer overflow, etc in a system that is often used in attacks on that system. The particular set of attack vectors is different for each system analyzed and the choice of attack vectors was not systematic, and was completely up to the discretion of the individual who was analyzing the system. In this early attack surface focused work typically, a security expert chose appropriate attack vectors to identify all possible attack points. Then these attack points were ranked in order of damage potential. Often in attack vector method an expert was also required to identify system’s known security flaws and issues along with security expert. Although attack vector method was effective for measuring the attack surface of a program, but it was not practical and could not be completed by an educated security novice, or completed systematically, such as through an analysis program. It required expert knowledge of both security and the program itself. The next method for measuring an attack surface adopted a qualitative approach. An input/output automaton was developed for a system, which allowed the easy identification of channels of attack and attackable resources .

Pratyusa K. Manadhata et al proposed a formal model for a system’s attack surface .They formalized the notion of a system’s attack surface using an I/O automata model of the system and defined a quantitative measure of the attack surface in terms of three kinds of resources used in attacks on the system: methods, channels, and data.In their research work they introduced the entry point and exit point framework based on an I/O automata model of a system and define the system’s attack surface in terms of the framework.They had also established that with respect to the same attacker, a larger attack surface of a system leads to a larger number of potential attacks on the system and introduced the notions of damage potential and effort to estimate a resource’s contribution to the measure of a system’s attack surface and define a qualitative and a quantitative measure of the attack surface.

III. MATERIALS AND METHODS

This study focuses solely on Java-based applications. Several open source java applications are selected to be the case studies for this research. We have chosen nine java applications (see appendix A) spanning to a wide range of software applications .

The Coupling Intensity values of each java applications are collected from the source code of above selected java applications[TABLE I] and analyzed based on the belief that what it measures is likely to have a significant(positive or negative) impact on an attack surface measurement of a system.

TABLE I
Coupling Intensity and Attack Surface for 9 Java Open-Source Programs

Java Applications	**Coupling intensity**	**surface Attack**
Bulles and Cows	4.68	24
Google2srt	3.72	32
HashcodeCracker	2.46	78
Jftp	4.27	104
JGPSTrackEdit	2.00	63
Memoranda	4.85	111
M ypassword	5.05	48
Scientific Calculator	7.06	13
Simple Video Convertor	5.06	27

The Microsoft Attack Surface Analyzer beta version[2] is used to measure the attack surface of the java applications. The analyzer scans for items such as running processes, open ports, new firewall rules, directories with weak permissions, and other unsecure events or states. The analyzer assigns a severity ranking to the security issues found, which represents the ratio of damage potential to effort value. After each application is scanned and a report is generated, the issues are totaled to get a measurement of the attack surface for each java application.

The correlation coefficients are calculated for each Coupling Intensity value and the attack surface as two data sets, yielding the correlation coefficient , which can be seen in Table II. Correlation coefficients range from -1 to 1. For correlations between two sets of data, a value of 0 means that no correlation whatsoever exists. If you have a value from one set, with a correlation of 0 you can make no assumptions at all about the corresponding value from the other set. If the correlation coefficient is positive, however, it means that as the values of one data set get larger so do the values in the other set. Similarly, if the correlation is negative, as values of one data set get larger, the values in the other data set will get smaller. A correlation coefficient of 1 or -1 means there is aperfect correlation between the two data sets; if the sets were graphed as a scatter plot, connecting the dots would yield a perfectly straight line.
Typically, a correlation coefficient from .85 to 1 or from -.85 to -1 is considered a strong correlation. In this context, it would mean that there is a strong correlation between a given metric and the attack surface size for this data set.

TABLE III
CORRELATION OF THE DESIGN METRICS AND ATTACK SURFACES

	Correlation Coefficient
Coupling Intensity	-0.4

RESULT ANALYSIS:

The first observation is that the attack surface is negatively correlates with Coupling Intensity.This result indicates that there is a weak negative correlation between coupling intensity and attack surface. Negative correlation indicates that when coupling intensity in a software design increases the future attack surface of the software product will decrease but it is contradictory as very high values of coupling intensity suggest that there is excessive coupling among operations, i.e., a sign that the calling operation does not “talk” with the right “counterpart”. Intuitively high values of coupling intensity must increase attack surface size of the application .Although Coupling Intensity better characterizes a software system w.r.t. coupling yet it is not strong predictor of software’s future attack surface size during design phase .But our result shows an average coupling intensity in a software design reduces the software’s future attack surface size. We can determine to find what the typical high and low values of coupling intensity by using statistical means as follows:

1. Select significant sample size of software systems(Java applications in our case)
2. Average (AVG), to determine the most typical value of the data set (i.e., the central tendency)
3. Standard deviation (STDEV), to get a measure of how much the value in the data set are spread.

Lower and higher margins of coupling intensity can be calculated as:
Lower margin: AVG − STDEV.
Higher margin: AVG + STDEV.
Very high: (AVG+STDEV) · 1.5, i.e., we consider a value to be very high if it is 50% higher than the threshold for a high value.

A value 2.46 is an average value of coupling intensity in our case which is the coupling intensity of HashcodeCracker. We can easily observe from the TABLE I that most of the java applications have higher coupling intensity except JGPSTrackEdit having value 2.00

T-Test Analysis

We conduct a t-test to determine whether the statistical relationship between the coupling intensity and the attack surface measure is real or just the result of chance. All of the t-tests are significant at the 0.05 level. The p-value of the T-test is listed in Table III.

TABLIIIII
T-Test P-Values Pairing Each Collected Metric with the Attack Surface Measurements

T-Test P-Values

	Two tailed P- Value
Coupling Intensity	***0.01***

To further investigate the findings, the scatter plots of Attack surface vs Coupling Intensity is examined in figure 1.

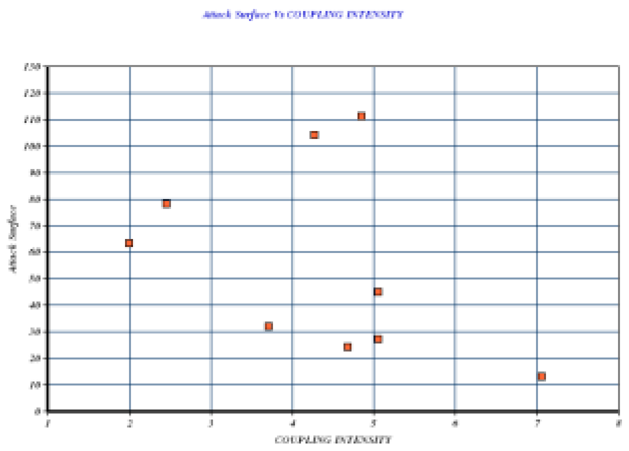

Fig 1: scatter graph of Attack Surface Vs Coupling Intensity

ISSN: 2278 – 7798
International Journal of Science, Engineering and Technology Research (IJSETR)
Volume 2, Issue 2, February 2013

CONCLUSION AND FUTURE WORK

In this research paper we have shown how coupling intensity is feasible predictors of system's future attack surface size. Considering all these metrics during the design process of any software will potentially lead to more secure applications.

In the future, research in this area should be taken in several different directions. First, other languages should be analyzed to determine if these java specific findings still hold true. Another avenue of research would be to specify the specific area in software design area that most contributes to attack surface.

REFERENCES

[1] Computer Emergency Response Team Coordination Center (CERT/CC), http://www.cert.org/stats/cert_stats.html (accessed July 2009).

.

[2] "Attack Surface Analyzer - Beta." Microsoft. Web. 18 Jan. 2011.

[3] <http://www.microsoft.com/download/en/details.aspx?id=19537>

[4] Manadhata, Pratyusa K. An Attack Surface Metric. Pittsburgh: Carnegie Mellon University, 2008.

[5] Howard, M., Pincus, J., Wing, J.: Measuring relative attack surfaces. In: Proc. of Workshop on Advanced Developments in Software and Systems Security (2003).

[6] Manadhata, P. andWing, J.: An Attack Surface Metric. IEEE Trans. Software Eng. 37, 371–386, 2011

[7] Howard, Michael. " Mitigate Security Risks by Minimizing the Code You Expose to Untrusted Users." http://msdn.microsoft.com/en-us/magazine/cc163882.aspx

[8] Hafiz, M., Adamczyk, P. and Johnson, R.: A Catalog of Security-Oriented Program Transformations. Technical Report UIUCDCS-R-2009-3031, University of Illinois at Urbana-Champaign, Jan 2009.

[9] Y. Shin, "Exploring Complexity Metrics as Indicators of Software Vulnerability," in Proceedings of the 3rd International Doctoral Symposium on Empirical Software *Engineering*, Kaiserslautem, Germany, Oct. 2008, available from the author's website http://www4.ncsu.edu/~yshin2/ papers /esem2008ds_shin.pdf (accessed July 2009).

[10] Y. Shin and L. Williams, "An Empirical Model to Predict Security Vulnerabilities Using Code Complexity Metrics," in Proceedings of the 2nd ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, Kaiserslautern, Germany, Oct. 2008, pp. 315-317.

APPENDIX A

Bulles and Cows: Classic mastermind game. http://sourceforge.net/projects/mybullsandcows/

Google2srt: Google2SRT is a tool that can download "not embedded" subtitles (Closed Captions - CC) from YouTube/Google Video videos (if those are present) and convert them to a standard format (SubRip - SRT) supported by most video players. http://google2srt.sourceforge.net

HashcodeCracker: This software cracks the MD5, SHA1,NTLM(Windows Password) hash codes.

Jftp: JFtp it is a FTP Client, written on java. Project provides well convenient user interface for performing ftp operations, generally for file or/and directories transfer. http://gftp.sourceforge.net/

JGPSTrackEdit: JGPSTrackEdit is a tool for editing gps tracks and planning (multiple days) tours (GPS track editor).. http://sourceforge.net/p/jgpstrackedit/wiki/Home/

Memoranda: Memoranda (formerly known as jNotes2) is a cross-platform diary manager and a personal project management tool. http://memoranda.sourceforge.net/

Mypassword: MyPasswords is a very light-weight, secure and easy-to-use password manager. http://www.mypasswords7.com/

Scientific Calculator : This Calculator is beautifully designed, having both standard and scientific modes of calculations. It is built to run anywhere, as its written completely in java. http://codecypher.blogspot.in/

Simple Video Convertor: FRONTEND for mencoder for Windows and Linux. Convert rmvb,avi,mp4,wmv, ogv, mkv, mov, mpg, vob,ogv, ogg, ogm etc To avi (divx5), xvid, dvd iso, h264 in a simple way. http://svideoconverter.sourceforge.net/