International Journal of Engineering Research
Volume No.5 Issue: Special 4, pp: 790-991
ISSN:2319-6890)(online),2347-5013(print)
20 May 2016

A Tool for Ferreting Out Software Vulnerability in Aegis & Armament Programs Redemption*

Prajna Bhavi[1], Dr. Chandramouli H[2], Dr. B.R.Prasad Babu[3],
MTech Student, Department of CSE ,R&D Centre, East Point College of Engg & Technology[1]
Professor, Department of CSE ,R & D Centre, East Point College of Engg & Technology[2]
Professor & Head of Department of CSE, East Point College of Engg & Technology[3]
[1]prajna.rb@gmail.com, [2]hcmcool123@gmail.com, 3brprasadbabu@gmail.com
*Project funded by the Government of Karnataka under VGST Scheme (2015-2016)

Abstract
Programming helplessness is a shortcoming that can be misused to access the code making the product exceedingly unstable. To make the product secure, vulnerabilities must be recognized and rectified. This Tool has been utilized to ascertain the Degree of Severity and finding remedy for it which helps developer.

KeyWords: Degree of Severity, Tool, Common Weakness Enumeration (CWE), Vulnerability Detection, Vulnerability Remedy, ISM

I.INTRODUCTION

Vulnerability threatened the security of the software at various stages in different stages, unintentionally because of mistakes made by developers or by deliberate piracy. Software design and implementation of the poor are the main causes of most security vulnerabilities threats [i]. Security issues may also occur from web sites and web applications (webapps). Need to be protected from all kind of threats and other data centres of the assets used to host Web sites and related systems.

Study aspects of the recent weakness evaluated on more than 250 web applications of e-commerce, online banking, enterprise collaboration, and supply chain management positions were held and found that 92% at least of Web applications vulnerable to the kind of pirate attacks.

It is doubtful whether the current information techniques for security will be able to protect critical software systems unless they make security an integral part of the program. Java language has emerged choose to build systems based on large and complex web, partly because of the safety language in which direct memory access and eliminate problems such as buffer overruns refused advantages?.

However in spite of these features, it is possible to make logical programming errors that lead to vulnerabilities such as SQL injection and cross-site scripting attacks [iii]. A simple programming error could be left vulnerable Web application for accessing unauthorized data and unauthorized updates, or delete data, and fall leading to denial of service attacks applications [iv]. Efforts should be made during the design and implementation of the program to make safe and protect software against it. This document discusses the vulnerabilities that are injected into the Java programs during the coding phase and describes a tool developed to detect vulnerabilities and warn the developer for these. Tool developed by the authors, reveals weaknesses in each program Java caused by off by one mistake, input validation is improper, Despicable check for unusual or exceptional conditions, deserialisation of untrusted data.

A. PRESENT STATE OF RESEARCH

Because of expanded episodes of data robbery, Security of programming is getting parcel of consideration, an affiliation which makes all the normal shortcoming and vulnerabilities .It causes being developed group to discover powerlessness in source code and in operational framework and administration of programming shortcoming. Various Tools are available in the market for static analysis of Java Code. The lists are

Checkstyle- This plug in works as the checkstyle rules. These rules tell where your code violates much like compiler but instead producing .class file, it produces warning. A violations will be reported. Checkstyle specifies which checks are validated against your code and with which severity [iv].

FindBugs-It is an open source from University of Maryland [v].

IntelliJidea - Cross-stage Java IE with own plan of a couple of hundred code examinations available for separating code on-the-fly in the mass examination.

JArchitect-Simplifies managing code by comparing different version of the code. This supports version control [vi]

PMD- It is static code analyser. It uses rule –set that define when a piece of source is erroneous [vii].

Of all these instruments accessible, apparatus created named Tool recognizes vulnerabilities in any Java program brought about by off by one mistake , input validation is improper, Despicable check for unusual or exceptional conditions,

doi : 10.17950/ijer/v5i4/034

ISSN:2319-6890)(online),2347-5013(print)
20 May 2016

deserialisation of untrusted data. Along with Degree of Severity metrics calculated.

II Material And Methodology

A. DEGREE OF SEVERITY IN A PROGRAM

Each of the weaknesses discussed in this paper has been assigned a severity level defined in CWE. In this paper we define a metric for calculating the Degree of Severity (referred to as ISM).

$$ISM = \sum_{I=1}^{m} Wi * Ni$$

Where,
ISM stands for the Degree of Severity,
i is the type of vulnerability where i=1,2,....m
Wi is the Severity of Vulnerability in the software
Ni is the frequency of occurrence of vulnerability i.

B.VARIOUS VULNERABILITIES WHEN & HOW THEY OCCUR

1. OFF BY ONE MISTAKE

This shortcoming will by and large prompt indistinct conduct and in this manner crashes. In case of loop index file variables, the probability of infinite loop is additionally high. A product figures or uses a mistaken value for minimum or maximum that is 1 more, or 1 less, than the right value [viii].

2 .INPUT VALIDATION IS IMPROPER

At the point when programming does not accept include appropriately, an assailant can make the info in a structure that is not expected by whatever is left of the application. This will prompt parts of the framework getting unintended data, which may bring about adjusted control stream, subjective control of an asset, or self-assertive code execution [ix].

3. DESPICABLE CHECK FOR UNUSUAL OR EXCEPTIONAL CONDITIONS

The programmer may assume that certain events or conditions will never occur or do not need to be worried about, such as low memory conditions, lack of access to resources due to restrictive permissions, or misbehaving clients or components. However, attackers may intentionally trigger these unusual conditions, thus violating the programmer's assumptions, possibly introducing Instability, incorrect behaviour, or vulnerability [x].

4. DESERIALIZATION OF UNTRUSTED DATA

It is often convenient to serialize objects for communication or to save them for later use. However, desterilized data or code can often be modified without using the provided accessory functions if it does not use cryptography to protect itself. Furthermore any cryptography would still be client-side security which is a dangerous security assumption. Data that is untrusted cannot be trusted to be well-formed [xi].

C.WORKING OF TOOL

The tool takes as input any Java program and scans to identify the vulnerabilities. If any vulnerability is detected then it displays warning message and suggests steps for its mitigation.

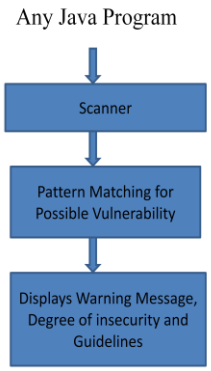

Figure 1 Working Procedure

The steps followed are:
1. Select the input Java program
2. Select from the drop down list all types of vulnerabilities intended to be detected
3.As shown figure As shown in Figure 1, for a Java program given as an input to the Tool, it displays type of vulnerability found and the place of its occurrence. It also gives the Degree of Severity in the input program

III RESULTS AND TABLES

A. OFF BY ONE MISTAKE

Common Observations

Affects Availability: This will cause undesired behaviour and system crash may happen, it may enter infinite loops

Affects Integrity: This modifies memory. Buffer overflow may occur.

Affects Confidentiality: Execute unauthorized code or command.

Remedy

When character arrays are copied, size of the parameter must be correct.

B.INPUT VALIDATION IS IMPROPER

Common Observations

doi : 10.17950/ijer/v5i4/034

ISSN:2319-6890)(online),2347-5013(print)
20 May 2016

Affects Availability: An intruder can provide unexpected value excessive usage of resource, system crash may occur.

Affects Confidentiality: An intruder can read confidential data

Affects Integrity: intruder may alter the control flow

Remedy:

Allow only whitelist variables which follow the strictly follow the specification during implementation.

C.DESPICABLE CHECK FOR UNUSUAL OR EXCEPTIONAL CONDITIONS

Common Observations

Affects Integrity: The information which were delivered as an after effect of a capacity call could be in a terrible state upon return. In the event that the arrival quality is not checked, then this awful information might be utilized as a part of operations, perhaps prompting an accident or other unintended practices.

Remedy:

Check return value of all the functions and verify the expected value.

D.DESERIALIZATION OF UNTRUSTED DATA

Common Observations

This varies by context. It depend on the objects and methods which are desterilized. This may modify the application data, cause unexpected state.

Remedy:

Define final readObject() to prevent deserialization explicitly

Table 1 Severity of Vulnerabilities

Type of Vulnerability	Severity
Off-by-one mistake	18
Input validation is improper	20
Despicable Check For Unusual OR Exceptional Condition	12
De-serialization of un-trusted data	7

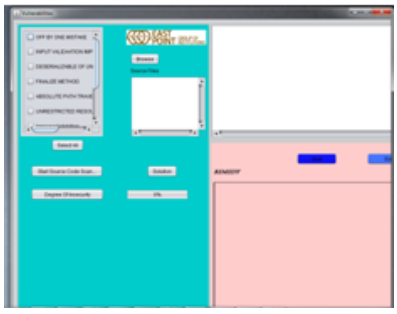

Figure 2. Working Tool Screenshot

IV.CONCLUSION

The tool described here detects vulnerabilities that exist in the code, calculates the degree of Severity of the input Java program and gives the remedy for that error.

The efficiency of the tool is designed to use for calculating the degree of uncertainty in two categories of programs: one written by experienced Java developers and the other students.

Acknowledgement

I would like to express my sincere thanks to ***Dr. B M Satish,*** Principal, East Point College of Engineering and Technology, ***Dr. B.R.Prasad Babu***, HOD, Department of CSE, ***Dr. Chandramouli H,*** Professor and PG Coordinator, Department of CSE for their guidance to complete technical paper work on time.

REFERENCE

[i] G. Mcgraw, Software Security: Building Security In, Addison Wesley, 2006.

[ii] A. K.Talukder, M. Chaitanya. Architecting Secure Software Systems, Auerbach Publications, 2009.

[iii] R. Priyadarshini, A. Basu and S. Sushma, "SecCheck: A Tool to Detect Vulnerabilities in Java Code," International Conference on On-Demand Computing, ICDOC Bangalore, Nov 15-16, 2012.

[iv] N. Ghosh and A. Basu, "WebCheck: A Tool to Detect Weaknesses in Java Web Applications," International Conference on Information and Communication Engineering ICICE Bangalore, June 28-29, 2013.

[v].findbugs.sourceforge.net/findbugs2.html

[vi].https://en.wikipedia.org/wiki/IntelliJ_IDEA

[vii].https://en.wikipedia.org/wiki/PMD_(Software)

[viii] Off by one error: http://cwe.mitre.org/data/definitions/193.html

[ix] Improper Input Validation: http://cwe.mitre.org/data/definitions/20.html

[x] Improper Check for unusual or exceptional conditions: http://cwe.mitre.org/data/definitions/754.html

[xi] Deserialization of Untrusted Data http://cwe.mitre.org/data/definitions/502.html

AUTHORS PROFILE

Ms.Prajna Bhavi is a MTech Student in Software Engineering in Department of Computer Science & Engineering, East Point College Of Engineering, Visvesvaraya Technological University. She has attended 2 Conference and presented 4 papers in National and International Conferences in various colleges and companies. Her research areas are Software Engineering, Testing, Development. Attended hands on workshop on Design Pattern held by CSI Chapter, Bangalore.

Dr.Chadramouli H is working as a professor, R&D Department of Computer Science & Engineering, East Point College of Engineering & Technology. His research areas are Wireless Sensor Networks, Mobile Adhoc Networks, IOT and Cloud Computing. He published more than 10 papers in various international Journals. Presently he is guiding for PhD Scholars in Visvesvaraya Technological University (VTU) India.

Dr.B.R.Prasad Babu is working as Professor and Head, Department of Computer Science and Engineering at East Point College of Engineering and Technology. His research areas are Mobile Adhoc Networks, Mobile Communication and Software Engineering. He published more than 50 papers in various international Journals. Presently he is guiding for PhD Scholars in Visvesvaraya Technological University (VTU) and Jawaharlal Nehru Technological University (JNTU)India.

doi : 10.17950/ijer/v5i4/034