***I.J. Education and Management Engineering*, 2018, 3, 14-20**
Published Online May 2018 in MECS (http://www.mecs-press.net)
DOI: 10.5815/ijeme.2018.03.02

Available online at http://www.mecs-press.net/ijeme

A Bug Tracking Tool for Efficient Penetration Testing

Kavita Pandey[a]

[a] *Department of CS&IT, Jaypee Institute of Information Technology, Noida, 201301, India*

Received: 30 October 2017; Accepted: 19 December 2017; Published: 08 May 2018

Abstract

A bug tracking system or defect tracking system is a software application that keeps track of reported software bugs in software development projects. A penetration test, colloquially known as a pen test, is an authorized simulated attack on a computer system that looks for security weaknesses. The use of the above two paradigms is very famous in the software development industry. Many researches have managed to expose some problems like manual testing, report generation, etc which the developers might face if these are not used properly. There is a plethora of software's that offer different kinds of solutions to the problems but somehow they lack one thing or the other. The objective here is to address these problems where the solution is using the bug tracking systems and penetration testing in an architectural manner. This would be done by using an already existing solution and then improvising upon it.

Index Terms: Bug, Tracker, Testing, Penetration, Collaboration, Software.

1. Introduction

A software bug is a flaw, error, or a fault in an approach or computer program that produces an inappropriate or unpredicted result. Maximum bugs evolve from mistakes and blunders made by people in a program's design or its source code. So, it becomes increasingly important for there to be a place to manage bugs so that an organization can work on them collaboratively.

By using bug tracking tools, we can keep track of testified software bugs with the assistance of bug tracking tools. A bug tracking tool is a software application which helps the computer programmer to keep track of all the reported software bugs in their task. Bugs can be of many types, ranging from quality, User Interface, server, to security ones. This is where the job of a tester becomes relevant. In software development teams many types of tests such as unit tests, integration tests, deployment tests, etc are performed as a part of the

* Corresponding author.
E-mail address:

development process [1] [2] [3].

Penetration testing can be used to tackle security weaknesses. It can also be used to test an organization's security policy compliance, its employees' security awareness and the organization's ability to identify and respond to security incidents [4] [5]. Penetration testing and bug tracker tools can be coupled together and put to good use when a security team uses the advantage of the latter. Some of the factors which shed light on the problems faced in real world software development process are as follows.

1.1. Need of a Bug Tracking Tool

The Software development companies face a lot of problems while manually maintaining their projects, bugs and status [6] [7]. This type of problem makes the whole system inefficient thus, giving way for a poor and unorganized process. In order to remove this problem a “Defect Tracking System” can be used which maintains a database of problem reports. Bug tracking software’s allow individuals or groups of developers to keep track of outstanding bugs in the product effectively. These softwares can track bugs and changes, communicate with members, submit and review patches, and manage quality assurance.

1.2. Effective Penetration Testing for Big Application

The organizations pay very close attention to potential security threats it faces during development [8]. A big software or an application can have hundreds or thousands of security issues that may need to be solved. This makes it seemingly obvious for the need of a tool that can generate many penetration test reports at the go for the software in a scalable manner.

1.3. Penetration test Reports for Different Bug Trackers

Nowadays, many Bug trackers are available for use. The penetration test reports can be reported to a bug tracker for other collaborators to review [9]. But a proper format isn’t maintained when it comes to managing the uniqueness of different bug trackers. A lot of currently available penetration testing resources lack report writing methodology. Such an approach can leads to a very big gap in the penetration testing cycle.

1.4. Problem of Manual Testing

Bug reporting involves tasks such as investigation, information gathering, testing and debugging throughout the whole process. It is very difficult to manage issues of a project manually, because hundreds of bugs can be found [10]. Software development cycle involves a number of iterations, where testing and reviewing are important parts of that cycle which must not hinder the overall process. Manual testing can prove to be quite hectic in such cases.

Once all the above problems would be addressed, a perfect product can be achieved for tackling security issues in a collaborative environment which can be used as a weapon in vulnerable situations. To address the problems, a tool is needed which can scan big applications at scale, and support many bug trackers. The rest of the paper is organized as follows. Section 2 covers the related work in this field while Section 3 provides a brief summary about how the problem is being tackled. It provides some insights into how the proposed work is useful and how it provides an edge. Lastly, Section 4 concludes the present exposition.

2. Related Work

There have been many types of software that make it possible to perform Penetration - Tests on web applications. Some of the popular ones are Kali Linux [11], Parosproxy [12], Wire shark, w3af, etc.

The operating system distribution known as Kali Linux, has been very popular among the hackers and crackers for exploiting security bugs and hacking into systems. This Operating System was originally designed for penetration testers. Although this can give an insight into the issues but it does not give a detailed report of the tests performed.

A Java based web proxy; called Parosproxy is used extensively to assess web application vulnerability. It supports editing and viewing HTTP/HTTPS messages on -the -fly to change items such as cookies and form fields. It includes a web traffic recorder, web spider, hash calculator, and a scanner for testing common web application attacks such as SQL injection and cross site scripting. It doesn't provide a framework for automated testing and so everything needs to be handled manually.

Simon Bennetts, a Security Engineer at Mozilla, inspired by Parosproxy, started working on a new project called OWASP Zed Attack Proxy. He decided to create a new project by forking Parosproxy and using it as a launching pad. ZAP provides all the features of Parosproxy along with many new features like automated testing, interception of HTTP requests while testing, etc. It provides support to a lot of users and is backed up by a big community. However this software needs to execute the idea of luring the software development team to use it for big projects.

The Bug Trackers today have provided many ways for developers to interact with their system through authentication mechanisms. Authentication is required in order to implement a tool where testers can work collaboratively with the team.

Table 1. Differences in the Bug Tracker usage

Bug Tracker Name	Custom Fields	REST API	SOAP	OAuth
Github	No	Yes	No	Yes
Bugzilla	Yes	Yes	Yes	No
JIRA	Yes	Yes	Yes	Yes

Table 1 shows that there are many different ways to interact with a Bug Tracker. Accordingly, the raised issues exclusively for one Bug Tracker whereas the other issues present in another Bug Tracker need to be modified. As mentioned above, these tools provide a solution to the some of the problems in question, like penetration testing, interacting with bug trackers, etc but don't tackle all of the problems. They lack some feature or the other therefore it becomes difficult to manage things in one place. Therefore, it was concluded that the software OWASP ZAP is a perfect base for developing a plugin which can use the new functionalities of ZAP. This will in turn facilitate a process in the Software development life cycle, which will enable the people able to work collaboratively on security threats and vulnerabilities. Sending detailed reports of the threat to the Bug Tracker repository would also be a part of the solution. The next section describes the implementation and technical details of the proposed solution.

3. Proposed Work

The OWASP Zed Attack Proxy (ZAP) is one of the world's most popular free security tools. It is actively maintained by hundreds of international volunteers. It can help users to automatically find security vulnerabilities in their web applications during development and testing phase of their applications. It is also a great tool for experienced penetration testers to use for manual security testing. It can scan in automatic mode too.

The main idea is to implement a Bug Tracker plugin for ZAP, so after integration the plugin will be able to communicate with the Bug Trackers and raise issues there itself. The tester can test the application, either manually or automatically and raise the issues by applying custom filters such as the priority of a bug, what risk does it cause, etc. This can be done by using either the ZAP GUI or the ZAP API, for developers trying to use the functionalities of ZAP externally.

The architecture in Figure 1 describes the implementation of the plugin as to how it fits in to the environment of ZAP. The Bug Tracker Add on, ExtensionBugTracker, will be extending the class ExtensionAdaptor. The API for this, BugTrackerAPI, will be extending the class ApiImplementor. The Semi -Automatic and the Automatic mode will have access to all the Alerts which will be fetched from the org.zaproxy.zap.extension.alert package as AlertNode objects via access to the AlertTreeModel. There will be another class "OptionsBugTracker" (extends AbstractParamPanel) for the above task. User will configure their Authentication Credentials which will ultimately be used by the Bug Tracker plugins (which implement the "BugTrackerIssue" class) and inject these details in BugTrackerIssue class which the add-on will use to send a request to the respective Bug Tracker API.

All the existing bug trackers today provide a different way of presenting the information of a bug besides some other differences. This extension provides a port for the Bug Tracker implementations to make them pluggable. If there is an implementation of any other Bug Tracker in the future no changes would be required in the Core Extension. Instead only the required methods must be implemented as provided in the class BugTracker along The ExtensionBugTracker provides the ability to add Bug Trackers. It also gets the already added Bug Trackers so the other Extensions of ZAP nondependent of this extension may be able to add more Bug Trackers as per need. Figure 2 explains the working of this solution. After opening ZAP, the user can go to the options screen and configure the Bug Trackers.

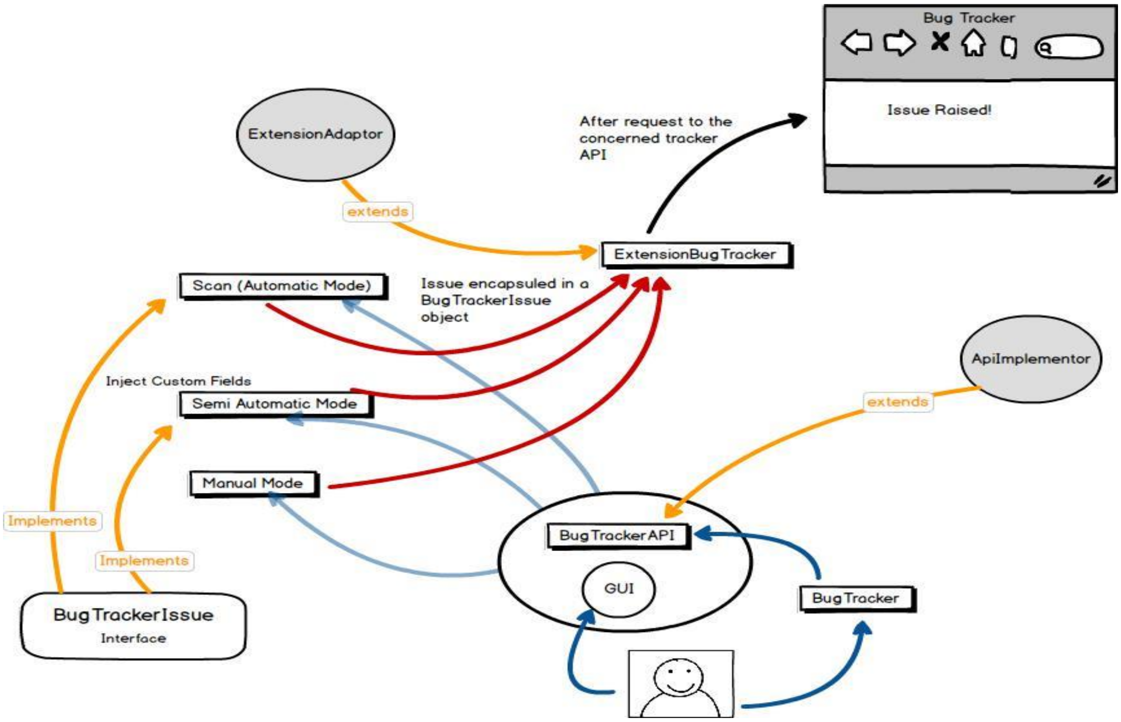


Fig.1. Architecture of the Bug Tracker

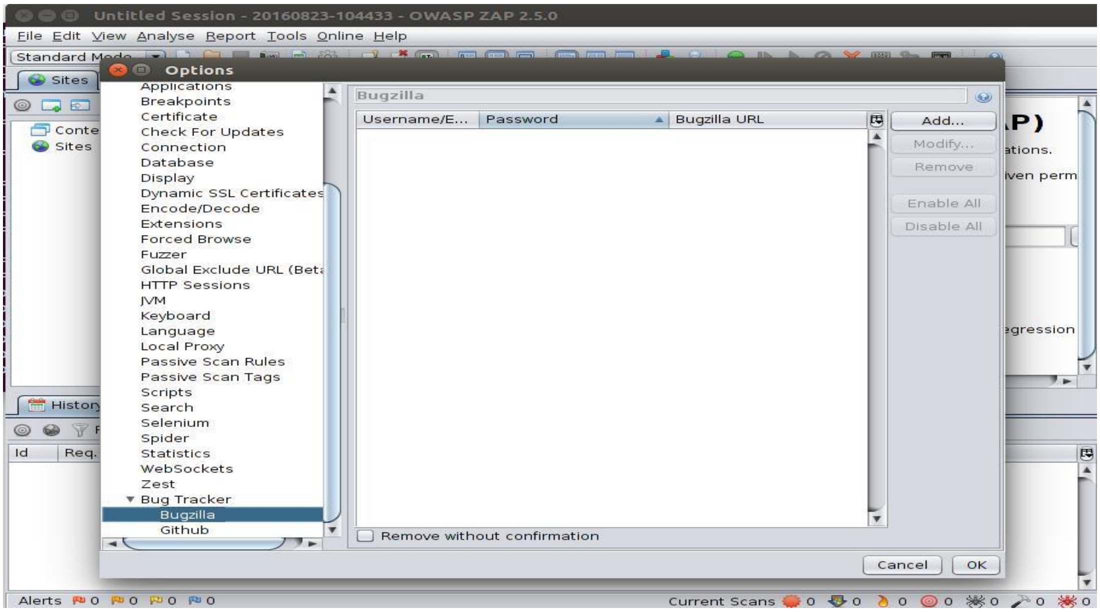
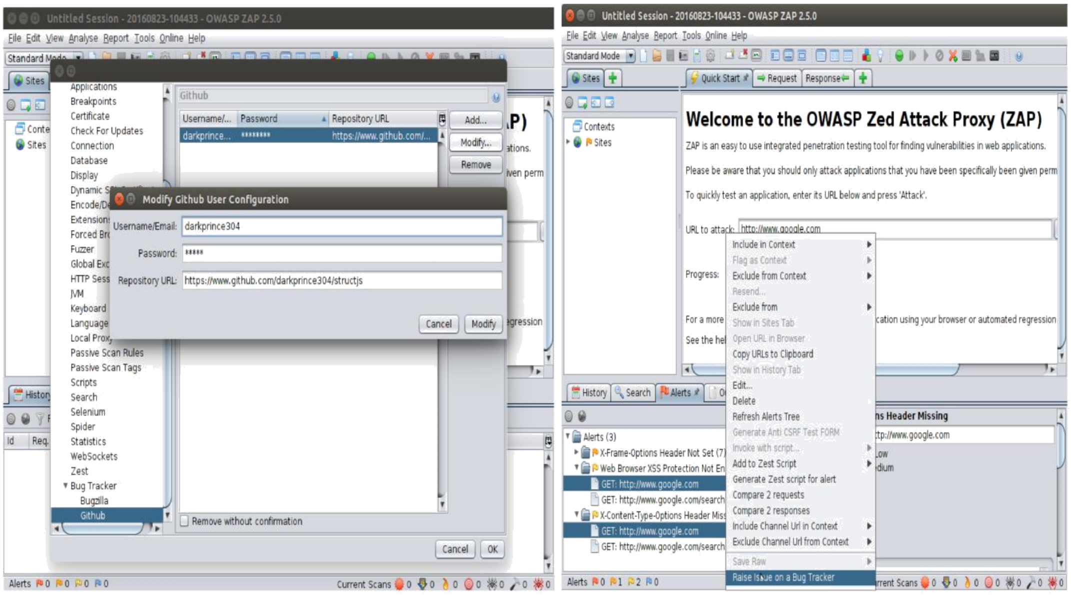


Fig.2. Choosing a Bug Tracker

The User can choose any Bug Tracker and then he or she can add, remove or modify its configurations as shown in Figure 3(a). The configurations include a login id (username or email), a password and a Bug Tracker field depending on the Tracker (values may depend on the bug tracker being used). These configurations would be used by the extension to show the tester a list of users which the ZAP user can use to authenticate with the bug tracker and raise an issue.


Modify Github User Configuration

Username/Email: darkprince304

Password: *****

Repository URL: https://www.github.com/darkprince304/structjs

Username/...	Password	Repository URL
darkprince...	********	https://www.github.com/...

Welcome to the OWASP Zed Attack Proxy (ZAP)

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web applications.

Please be aware that you should only attack applications that you have been specifically been given perm

To quickly test an application, enter its URL below and press 'Attack'.

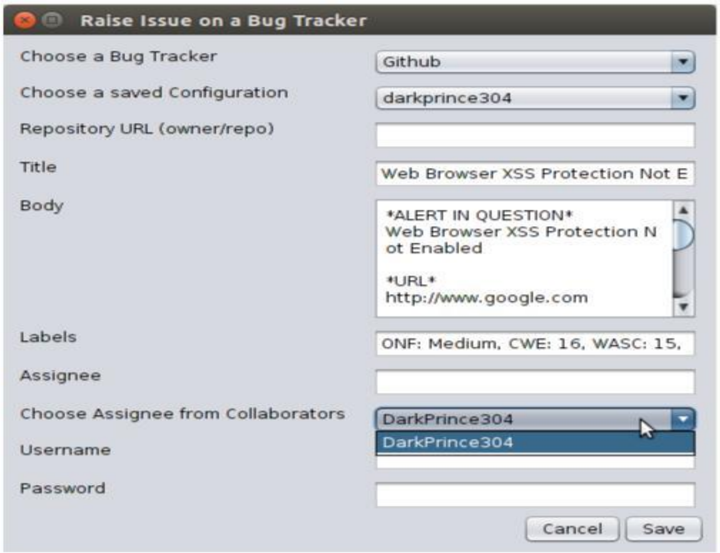
URL to attack: http://www.google.com

Untitled Session - 20160823-104433 - OWASP ZAP 2.5.0
File Edit View Analyse Report Tools Online Help
Standard Mode
Github
Add...
Modify...
Remove
Remove without confirmation
Bug Tracker
Bugzilla
Github
Cancel
Modify
Cancel
OK
Sites
Quick Start
Request
Response
Contexts
Sites
Progress:
History
Search
Alerts
Alerts (3)
X-Frame-Options Header Not Set (7)
Web Browser XSS Protection Not En
GET: http://www.google.com
GET: http://www.google.com/search
X-Content-Type-Options Header Miss
GET: http://www.google.com
GET: http://www.google.com/search
Include in Context
Flag as Context
Exclude from Context
Resend...
Exclude from
Show in Sites Tab
Open URL in Browser
Copy URLs to Clipboard
Show in History Tab
Edit...
Delete
Refresh Alerts Tree
Generate Anti CSRF Test FORM
Invoke with script...
Add to Zest Script
Generate Zest script for alert
Compare 2 requests
Compare 2 responses
Include Channel Url in Context
Exclude Channel Url from Context
Save Raw
Raise Issue on a Bug Tracker
Alerts 0 1 2 0
Current Scans 0 0 0 0 0 0 0

Fig.3. (a) Adding a Configuration (b) Raising in Semi Automatic Mode

This Bug Tracker Extension works in two modes: Semi Automatic and Automatic. In Semi Automatic Mode, the user must scan a web application using the "Scan" functionality of ZAP. When scanned ZAP will show a list of alerts to the user. The user can choose an alert directly from the node or from the sub node of vulnerability as shown in the Figure 3 (b).

When clicked "Raise Issue on a Bug Tracker" a dialog will pop up with a list of Bug Trackers for the user to choose to raise the issue in along with the appropriate Bug Tracker Fields. The screenshot below shows Github chosen and also shows the list of users added in the configurations (in the Options Screen). The user may choose any configuration and accordingly the assignee list will refresh with the collaborators of the repository as shown in Figure 4.



Raise Issue on a Bug Tracker

Choose a Bug Tracker: Github

Choose a saved Configuration: darkprince304

Repository URL (owner/repo):

Title: Web Browser XSS Protection Not E

Body:

```
*ALERT IN QUESTION*
Web Browser XSS Protection N
ot Enabled

*URL*
http://www.google.com
```

Labels: ONF: Medium, CWE: 16, WASC: 15,

Assignee:

Choose Assignee from Collaborators: DarkPrince304

DarkPrince304

Username:

Password:

Fig.4. Choosing a Configuration and Providing Detailed Report

The user can either choose a configuration or specify the repository explicitly in the Repo URL field. These steps give a good idea how this solution provides an easy and simple to use solution for automated penetration testing with detailed reports and high customizability.

4. Conclusion

In the present exposition, the Penetration Testing Software OWASP ZAP can be used to address the following problems; Need to report test reports to a bug tracker, testing of big applications, generation of concise penetration test reports, and automatic penetration testing of web apps. This solution improvises upon the solution provided by Kali Linux by reporting the test results on a bug tracker and in a detailed manner. It also provides a framework for the implementation of future Bug Trackers and leaves the implementation details to the developers working on it. The API for doing this has been exposed through the framework. The framework also provides functionality for automated testing by providing flexibility in raising the issues according to the needs of the tester. This was something that was lacking in Parosproxy and worked as an inspiration for this project. The solution provides a wrapper around the testing stage in Software Development Lifecycle. This is done by enabling the developers to test and work on their applications by using the advantages that the Bug Trackers and Penetration Testing provide.

References

[1] Khulood Salem Albeladi, M. Rizwan Jameel Qureshi, "Improvement of Component Integration Testing Technique", IJITCS, vol.5, no.8, pp.109-122, 2013. DOI: 10.5815/ijitcs.2013.08.11

[2] Mahmood H., Sirshar M., "A Case Study of Web Based Application by Analyzing Performance of a Testing tool", I.J. Education and Management Engineering, Vol. 4, pp. 51-58, 2017.

[3] Abhinandan H. Patil, Neena Goveas, Krishnan Rangarajan, "Regression Test Suite Prioritization using Residual Test Coverage Algorithm and Statistical Techniques", International Journal of Education and Management Engineering(IJEME), Vol.6, No.5, pp.32-39, 2016.DOI: 10.5815/ijeme.2016.05.04

[4] Mohd. Anjum, Md. Asraful Haque, Nesar Ahmad, "Analysis and Ranking of Software Reliability Models based on weighted criteria value", vol. 2, pp. 1-14, 2013.

[5] Tang Hong, Fan Guo-ling, "Based on The Research and Application of Tengen Software in Architectural Design Teaching Process", IJEME, vol.1, no.5, pp.26-31, 2011.

[6] A.S. Syed Fiaz, N. Devi, S. Aarthi, "Bug Tracking and Reporting System" in International Journal of Soft Computing and Engineering (IJSCE)ISSN: 2231 -2307, Volume -3, Issue -1, March 2013.

[7] Zhang Gang, Wang Xiaolin, "Conformity Degree Analysis on Software Engineering Program and Professional Norms", IJEME, vol.3, no.2, pp.72-76, 2013.

[8] Mansour A. Alharbi, "Writing a Penetration Testing Report", April 6th 2010.

[9] Kanaklata and Shweta Sharma, "Survey and Study of various Bug Tracking and Logging Toolkits" in International Journal of Computer Applications (0975–8887)Volume 116–No.12, April 2015.

[10] Sandeep Singh, "Analysis of Bug Tracking Tools" in International Journal of Scientific & Engineering Research, Volume 4, Issue 7, July -2013, ISSN 2229 -5518.

[11] Suraj S. Mandalik, "Penetration Testing: An Art of Securing the System (Using Kali Linux)" in International Journal of Advanced Research in Computer Science and Software Engineering, Volume 5, Issue 10, October 2015, ISSN: 2277 128X

[12] Kali Linux | Penetration Testing and Ethical Hacking Linux Distribution

[13] Parosproxy -SecTools Top Network Security Tools.

Authors' Profiles

Kavita Pandey received her B.Tech. in Computer Science and Engineering from M.D. University in 2002 and M.Tech. (CS) from Banasthali Vidyapeeth University in year 2003. She has obtained her Ph.D. (CS) from Jaypee Institute of Information Technology, Noida, India in January 2017. She is currently working as an Assistant Professor in JIIT, Noida. Her research interests include Mobile Ad hoc Networks, Vehicular Ad hoc Networks, Optimization Techniques and Network Security. She has published various papers in International journals and conferences including Wiley, IEEE, Springer, Inderscience, etc.

How to cite this paper: Kavita Pandey,"A Bug Tracking Tool for Efficient Penetration Testing", International Journal of Education and Management Engineering(IJEME), Vol.8, No.3, pp.14-20, 2018.DOI: 10.5815/ijeme.2018.03.02