Jing Song U65712216

Professor Osama Alshaykh

EC601 Product Design ECE

Mini Project 2 Cyber Security Report

October 20, 2019

Pros and Cons of Open-Source Security Framework and Its Applicable Usage

In 2008, Apache Shiro, a Java open-source security framework was born. “Shiro” means “castle” in Japanese, and the framework was built to provide a “simple but powerful” way to secure the web applications. We could now use Apache Shiro API to realize functions like authentication, authorization, cryptography, session management, etc. for our own applications. These four features are the main usages that developers demand from such a security framework. Authentication makes sure that certain functions are only accessible for login users. Authorization allows developers to control user access. Cryptography could hide personal and sensitive data from the public, and session management allows each user to have his or her own “time sensitive state” (Hazlewood). Apache Shiro is just one of the many open-source security frameworks available on the Internet that allow developers to secure their applications. These security tools do make life much easier for developers, but we should still be aware of some drawbacks they will possibly bring us.

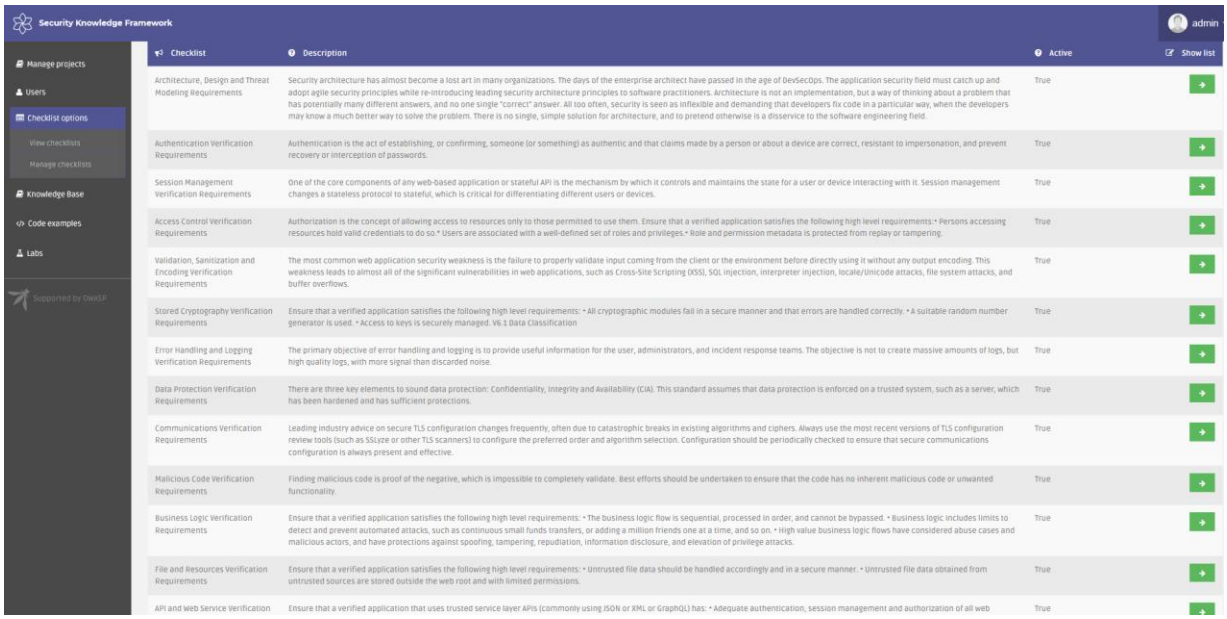
OWASP (the Open Web Application Security Project) is an organization and a community

where lots of security developers gather to work on projects that improve application security. The organization is non-profit and it allow any individual or organization to access it ("Main Page"). The people in this community has already developed a quite mature security framework and it is licensed under the "Creative Commons Attribution-ShareAlike 3.0 license" ("OWASP Open Cyber Security Framework Project"). Open-source security frameworks like the one provided by OWASP saves tons of time for developers. Instead of figuring out a complicated security implementation on their own, developers now could adopt these frameworks into their own softwares, sometimes even without any cost. Developing such framework in an open community also makes it easier to find out bugs and fix them. Since "developers care about their reputations", they would work very diligently to find potential problems in their code and show off the perfect code ("Three Myths Debunked About Open Source Software Security."). People who are paid to develop such a security framework has an incentive to find all flaws in the code, while those who work on non-profit projects may still be self-motivated to perfect their code. Either way makes open-source security framework better to adapt into softwares.

Even though many developers are working on fixing bugs in these open-source security frameworks, problems may still arise from all different aspects. For instance, fixing security problems require expertise in this area, and not all developers master such expertise. Those who do not may even give misleading ideas, so that the bugs would be not fixed or could be even worse. Besides, open-source security framework is open to the public and many companies or organizations may have already been using it. It is possible that people with bad intensions find a bug in the open-source code and use it to hack the existing security system. A

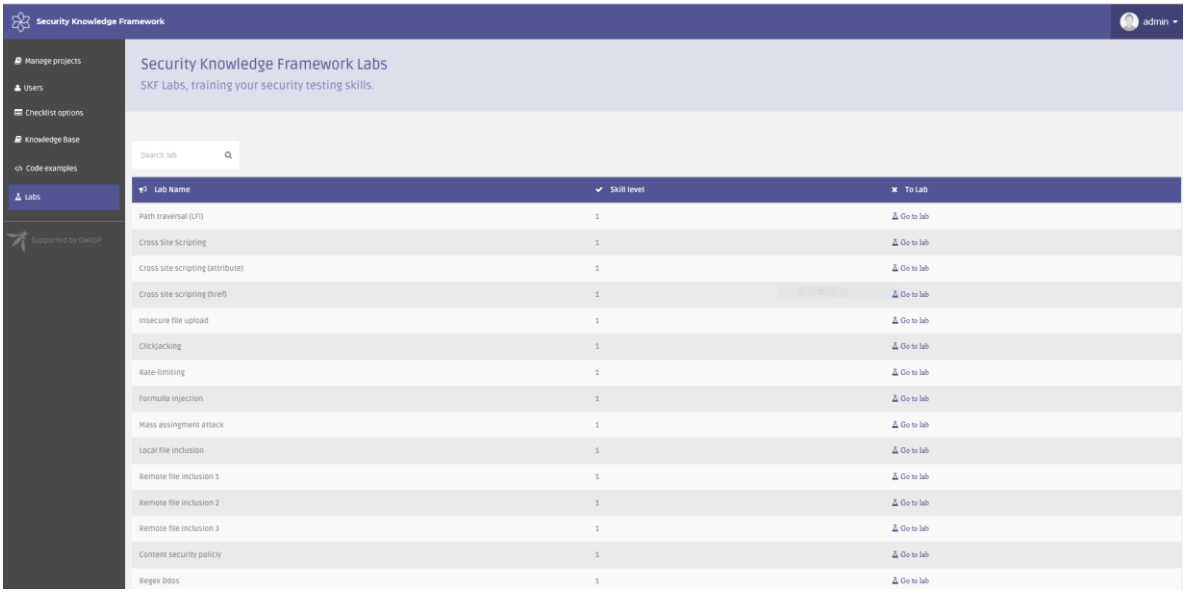
short time frame between one finding a bug and another person finding a bug could lead to unexpected results. An open-source security framework has to balance the publicity of code and being really secure.

OWASP also provides a security knowledge framework as a guideline for developers to implement the security requirements for their softwares. The “checklist” option of the framework looks like this on the website:



Screenshot from https://demo.securityknowledgeframework.org/checklist

The knowledge framework has a “lab” option for developers to try out several security testings on their own. Such an interactive approach would be very useful for beginners in cybersecurity and I would recommend anyone who is interested to try out this page.



Screenshot from https://demo.securityknowledgeframework.org/labs

Overall, open-source security framework is a good resource for developers if they need some guidelines on the security implementation or would like to save time and cost. However, these open-source security framework does not guarantee 100% security. Cyber security is such a big topic and we could not expect to solve all security problems with simply adopting the open-source security frameworks.

Work Cited

Hazlewood, Les. “Application Security With Apache Shiro.” *InfoQ*, InfoQ, 14 Mar. 2011, https://www.infoq.com/articles/apache-shiro/.

“Main Page.” *OWASP*, https://www.owasp.org/index.php/Main_Page.

“OWASP Open Cyber Security Framework Project.” *OWASP*, https://www.owasp.org/index.php/OWASP_Open_Cyber_Security_Framework_Project.

“Three Myths Debunked About Open Source Software Security.” *RubyGarage*, https://rubygarage.org/blog/open-source-software-security.

Summary of Yi-Wei Chen's Report(https://github.com/ThomasChen1997/EC601-Mini-Project-2/blob/master/Mini-project%202.pdf):

Yi-Wei Chen describes in details the purpose of cyberattacks and how different groups of people could defend themselves from cyberattacks.

Purposes of Cyberattacks:

(1) Damage sensitive info

(2) Extorting money

(3) Interrupt business activities

How to protect oneself from cyberattacks:

(1) Users: strong password, phishing emails, back-up data

(2) Organizations/ Companies: reliable security framework

(3) Technology: protect endpoint devices, networks, cloud

He also talks about the four most common types of attacks:

Four types of attacks:

(1) Ransomware: malicious software asking for ransom payment

(2) Malware: unauthorized access to damage the computer

(3) Social Engineering: tricks users to reveal sensitive information

(4) (most common) Phishing: fraud emails

Then he introduces the open-source security framework: Clusterfuzz, which is very helpful for software companies to locate security issues in their software products. Yi-Wei gives a list of some nice features of Clusterfuzz:

1. Highly scalable. Google's internal instance runs on over 25,000 machines.
2. Accurate deduplication of crashes.
3. Fully automatic bug filing and closing for issue trackers (Monorail only for now).
4. Testcase minimization.
5. Regression finding through bisection.
6. Statistics for analyzing fuzzer performance, and crash rates.
7. Easy-to-use web interface for management and viewing crashes.
8. Firebase authentication.
9. Support for coverage guided fuzzing (e.g. libFuzzer and AFL) and blackbox fuzzing.
10. Learners can get the new updates, versions or suggestions from Github.
11. It is operated in Python which contains lots of library to use.

Clusterfuzz has two main components, one is called App Engine, which provides a web user interface for users to access the stats and info on crashes. The other part is a pool of bots, these bots run preemptible (shuts down the machine) and non-preemptible tasks (does not shut down the machine). Users could choose between the different types of tasks, depending on their needs.

What I learned from Yi-Wei's paper is that our reports could be read by anyone from all different backgrounds, so it is always necessary to introduce the topic and include some basic concepts of cybersecurity in the report, so that our reader could understand the concepts better and follow closely the further discussion of cybersecurity in the report. Besides, I did not list enough detailed features of the open-source cybersecurity frameworks in my report and I should not have missed writing about them.