

Integration of Static Security Code Analysis in Continuous Integration Lifecycles

Sebastian Funke, Brian Pfretzschner, Hamza Zulfiqar
Center for Advanced Security Research Darmstadt
Department of Computer Science
Technische Universität Darmstadt, Germany

Abstract—In our paper we present our research results for the question: How to integrate static code analysis for security in a common continuous integration (CI) process of software development. And evaluate the vulnerability reporting capabilities in Jenkins and the open source quality management tool SonarQube. We used the popular CI tool Jenkins on a NIST standardized C test project¹ with a variety of vulnerabilities. Thereby we included a couple of static analysis tools in Jenkins for finding bugs and vulnerabilities during build and after the build process. Furthermore we analyzed how input validation is deployed in popular PHP frameworks.

I. INTRODUCTION

Content from our slides...about input validation and how static code analysis works. Limitation on open source, static analysis tools. Jenkins, SonarQube...and why [1]. Content of our paper, what comes when blabla...

II. INPUT VALIDATION IN POPULAR FRAMEWORKS

Check those <http://codegeekz.com/20-best-php-frameworks-developers-august-2014/> and make a table how input validation is handled there...

III. STATIC CODE ANALYSIS IN JENKINS

Used test suite: wireshark 1.8 from NIST testsuites with 85 vulnerabilities. Because its C, because its one of the most security critical languages and there are many good analyzers for C. Why not Juliet TestSuite with 65 000 vulnerabilities? Because they are collections of testcases and not a easily buildable project and the analysis and build process would take to long to evaluate the reporting features of the analyzers.

Todo: Table with tools with pros and cons

IV. STATIC CODE ANALYSIS IN SONARQUBE

V. EVALUATION OF REPORTING CAPABILITIES

Definition for userfriendly vulnerability reporting needed! Metrics for evaluation of reports needed and need to be mapped on the useability definition.

¹<http://samate.nist.gov/SRD/testsuite.php>

Installiere Plugins/Aktualisierungen

Vorbereitung

- Überprüfe Zugang zum Internet
- Überprüfe Zugang zu jenkins-ci.org-Server
- Erfolgreich

Coverity Plugin	Erfolgreich
Fortify 360 Plugin	Erfolgreich
Kiuwan Plugin	Erfolgreich
Static Code Analysis Plug-ins	Erfolgreich
FindBugs Plugin	Erfolgreich
Checkstyle Plugin	Erfolgreich
Android Lint Plugin	Erfolgreich
PMD Plugin	Erfolgreich
PRQA Plugin	Erfolgreich
ChuckNorris Plugin	Erfolgreich
CodeScanner Plugin	Erfolgreich
Brakeman Plugin	Erfolgreich
OWASP Dependency-Check Plugin	Erfolgreich
Analysis Collector Plugin	Erfolgreich
Cppcheck Plugin	Erfolgreich
Violations	Erfolgreich
Static Code Analysis Plug-ins	analysis-core Plugin war schon installiert. Jenkins aktiv wird.
FxCop Runner Plugin	Erfolgreich
DRY Plugin	Wird installiert
Jenkins neu starten	Wartet

[Zurück zur Startseite](#)
(Sie können die installierten Plugins sofort verwenden)

Fig. 1. Just a few used Jenkins plugins for static code analysis

A. Jenkins

B. SonarQube

VI. CONCLUSION

- Conclusion about how input validation is done in frameworks, what can be better ...
- Conclusion, is it better to integrate static analysis in Jenkins or just use SonarQubeits really not that

easy to find the right static code analyzer for your project with a specific programming language. There are lots of open source tools, but very old and just supported by Jenkins over hacks.

- Conclusion, is reporting in Jenkins useable

VII. SOURCES AND USEFUL LINKS

- Building Security In Using Continuous Integration
- TeamCity
Supports static code analysis. Also, it can import analysis reports produced by these tools: PMD, PMD/CPD, FindBugs, Checkstyle or JSLint.
- Jenkins
- Apache Continuum

REFERENCES

- [1] Consultative Committee for Space Data Systems (CCSDS), "Telemetry channel coding," *Blue Book*, no. 4, 1999. [Online]. Available: <http://www.ccsds.org/documents/pdf/CCSDS-101.0-B-4.pdf>