Static Code Analysis for Security in Continuous Integration

Sebastian Funke
Secure Software Engineering
TU Darmstadt
sebastian.funke@stud.tu-darmstadt.de

Brian Pfretzschner
Secure Software Engineering
TU Darmstadt
brian.pfretzschner@stud.tu-darmstadt.de

Hamza Zulfiqar
Secure Software Engineering
TU Darmstadt
hamza.zulfiqar@stud.tu-darmstadt.de

***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 project[1] 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

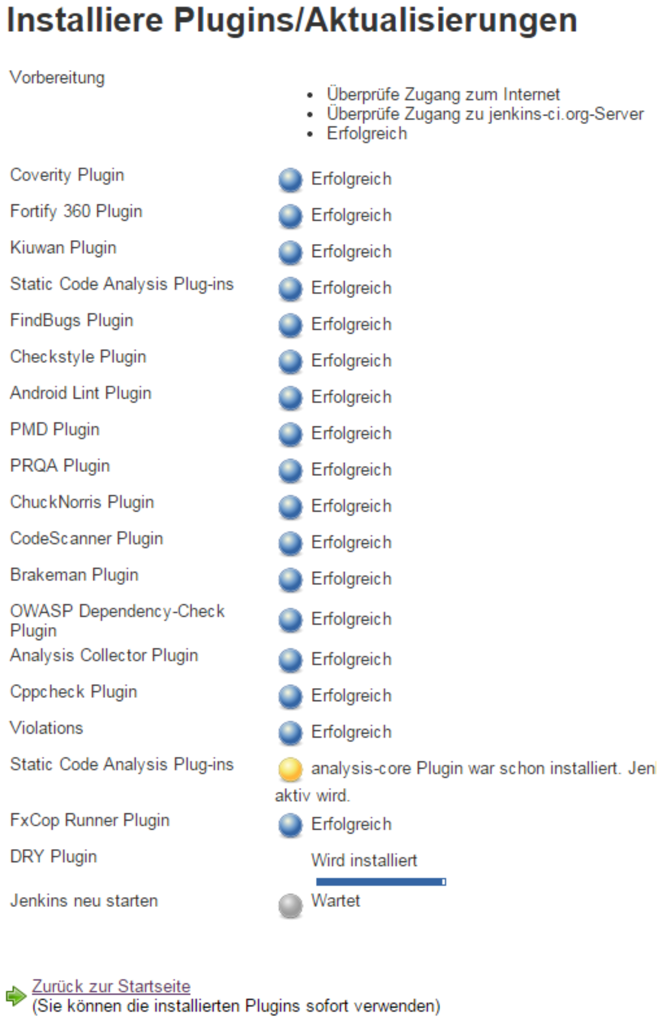
Used test suite: wireshark 1.8 from NIST testsuites with 85 vulnerabilities. Because its C, because its one of the most security critical languages and there are many good analyzers for C. Why not Juliet TestSuite with 65 000 vulnerabilities? Because they are collections of testcases and not a easily 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 useabillity definition.



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. Jen 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

[1] http://samate.nist.gov/SRD/testsuite.php

Jenkins or just use SonarQube ....its really not that easy to find the right static code analyzer for your project with a specific programming language.

- Conclusion, is reporting in Jenkins useable

REFERENCES

[1] Consulative 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