Poster: IDE Plugins for Secure Coding

Aniqua Z. Baset
University of Utah
aniqua@cs.utah.edu

Tamara Denning
University of Utah
tdenning@cs.utah.edu

***Abstract*—Many vulnerabilities in products and systems could be avoided if better secure coding practices were in place. There exist a number of Integrated Development Environment (IDE) plugins which help developers check for security flaws while they code. In this work, we present a review of such plugins. We believe that this work lays the groundwork for future research in this area by synthesizing the information necessary to orient researchers choosing to tackle this underexplored space.**

I. Introduction

Many vulnerabilities in today's consumer products result from common, well-documented coding errors. These weaknesses have been well-known for many years; however, they can still regularly be found in new systems. This is not necessarily a reflection on developers' intentions or training. After all, security is rarely a developer's primary task and there are many ways to introduce security flaws; for example, there are 471 vulnerability listings in CWE (Common Weakness Enumeration) that correspond to types of buffer overread and overwrite errors [1]. The general situation is exacerbated by the fact that many apps and Internet-of-Things devices are now developed by smaller, newer companies that might have less of an infrastructure for checking code security than larger companies with a longer history of code development.

Both static and dynamic analysis tools have been developed to detect security flaws in code (e.g., [2], [3], [4], [5]). These tools normally come with their own command-line or graphical interfaces to run analyses and display results. This requires developers to move back and forth between their coding environment (e.g., IDE) where they program, and the tool's interface, where they separately check for security problems. This overhead oftentimes contributes to lower adoption of security tools [6]. In recent years static analysis for security has become available via IDE plugins, providing a more seamless experience. These plugins allow developers to check security flaws in their code from within their IDE, since they present their results in the IDE like regular compiler errors. This in-situ security analysis and feedback can help developers detect flaws in the earlier stages of software development. In this work we synthesize information on such security IDE plugins.

II. Security Plugins in the Wild

We gathered security plugin information in four ways. First, we searched the plugin lists and marketplaces for four of the most prominent IDEs: Eclipse, IntelliJ IDEA, Visual Studio, and Netbeans IDEs. Second, we looked for plugins in forum discussions like StackExchange. Third, we checked lists of static security analysis tools (e.g., [7], [8]) to determine whether any of them have support for IDE integration. Fourth, we searched for security plugins developed in the academic literature.

We list the available IDE security plugins in Table I. We exclude some IDE plugins from our list that do not present results within the IDE. For example, the Eclipse plugin for Coverity uploads the code to a server; once the server-side analysis is complete the result is presented via the developer's online account. In contrast, we *do* include Checkmarx CxSAST, Fortify, and Veracode: while the analysis is performed on a server, the results are presented in the IDE similar to the other listed plugins. We also exclude Contrast since the Eclipse plugin version of it has been discontinued [9].

As evident from Table I, security plugins are available for most mainstream IDEs and languages/platforms, with the partial exception of Ruby and Android. We find only two plugins for Ruby (Checkmarx CxSAST, Veracode) and among all the plugins only Lint and FindBugs are available for Android Studio. We failed to find plugins for text-based editors such as Vim and Sublime.

Among the plugin listings we encountered for different IDEs, only the Eclipse marketplace reports on number of installations. Considering the installation numbers and their ranks in the marketplace, security plugins do not seem as popular as security researchers might hope. Only exception is Findbugs, it has very high installation numbers and is ranked #13 in the Eclipse marketplace in terms of installation numbers. We posit that this high popularity is due to the variety of features it offers beyond input-related vulnerability checking.

We have observed differences in quality and thoroughness in analysis reporting among plugins. Some plugins provide details in their report such as possible attacks, how the problem in code can lead to those attacks, examples of vulnerable and secure code, and risk ratings. Other plugins only mention the name of the possible attack or provide brief description of the attack. Besides pointing out the problem areas, some plugins also suggest possible mitigation strategies. However, in most cases these detailed reporting techniques serve to educate the developer on the identified attack and are not quick fixes specific to the code. To provide flexibility, some plugins also allow users to temporarily turn off particular warnings or to select/unselect specific vulnerability checks. Prior research suggests that such customization options make plugins more

TABLE I: IDE plugins available for security checks

Plugin	IDE	Language and/or Platform	Availability	Source	Introduced	Last update
Android Lint [10]	AS, Eclipse	Java, XML, Android	Free	Open	—	—
ASIDE [11], [12]	Eclipse	Java, PHP	Free	Open	Feb'13	Sept'14
CodeDX* [13]	Eclipse, VS	Java, .NET, Android	Commercial	Closed	Jan'15	Feb/Mar'16
Codepro AnalytiX [14]	Eclipse	Java, JSP, XML	Free	—	Feb'05	Oct'10
Cppcheclipse [15]	Eclipse	C/C++	Free	Open	Oct'09	Feb'16
Checkmarx CxSAST§ [16]	Eclipse, VS, IntelliJ	Java, .NET, Python, Ruby, C/C++, C#, JS	Commercial	Closed	—	—
ESVD [17], [18]	Eclipse	Java	Free	Closed	July14	Nov'16
Findbugs [19]	Eclipse, NB, IntelliJ, AS	Java, Android	Free	Open	—	—
Fortify [20]	Eclipse, VS	C/C++, Java, .NET, PHP, JS, Python	Commercial	Closed	—	Feb/Mar'17
FxCop [21]	VS	.NET	Free	Closed	—	—
Goanna Studio [22]	Eclipse, VS	C/C++	Commercial	Closed	—	—
Klocwork Insight‡ [23]	Eclipse, IntelliJ, VS	Java, C/C++, C#	Commercial	Closed	—	—
LAPSE+ [24], [25]	Eclipse	Java	Free	Open	Mar'11	Mar'11
SecureAssist [26]	Eclipse, VS, IntelliJ	Java, PHP, .NET	Commercial	—	—	—
SensioLabsInsight [27]	PHPStorm	PHP	Both	Closed	Oct'14	Jan'17
SonarLint [28]	Eclipse, VS, IntelliJ	Java, JS, PHP, .NET, Python	Free	Open	Oct'15	Feb'17
SSVChecker* [29], [30]	Eclipse	C/C++, Python, PHP	Free	Closed	May'10	Nov'16
Veracode [31]	Eclipse, VS, IntelliJ	Java, C/C++, C#, .NET, Python, Ruby, JS, PHP, Android	Commercial	Closed	—	Feb'17

VS = Visual Studio, IntelliJ = IntelliJ IDEA, NB = NetBeans, AS = Android Studio, JS = JavaScript
**Runs multiple analysis tools and present the combined results, §Previous version: CxSuite, ‡Previous version: Klocwork Solo*
ASIDE, ESVD, LAPSE+, and SSVChecker are academic. The standalone version of Findbugs is also from academic work [32].

usable [6].

III. Conclusion

IDE plugins that check for vulnerabilities can help increase the security of code. We find that there is a lack of information on these plugins about specific vulnerability checks and detection accuracy, which may contribute to lower adoption among developers. In addition to more complete information, we would like for security benchmarking information to be made available for each plugin so that the developer and security communities at large can better evaluate such plugins.

References

[1] "Cisco 2015 midyear security report. http://www.cisco.com/assets/global/UK/events/switchup_challenge/pdf/cisco-msr-2015.pdf."

[2] "Coverity: Static code analysis. https://www.synopsys.com/software-integrity/products/static-code-analysis.html#."

[3] "Cppcheck. http://cppcheck.sourceforge.net/."

[4] "Purifyplus. http://teamblue.unicomsi.com/products/purifyplus/."

[5] "Valgrind. http://valgrind.org/."

[6] B. Johnson, Y. Song, E. Murphy-Hill, and R. Bowdidge, "Why don't software developers use static analysis tools to find bugs?" in *Software Engineering (ICSE), 2013 35th International Conference on*. IEEE, 2013, pp. 672–681.

[7] "List of tools for static code analysis. https://en.wikipedia.org/wiki/List_of_tools_for_static_code_analysi."

[8] "Owasp: Static code analysis. https://www.owasp.org/index.php/Static_Code_Analysis."

[9] "Contrast. https://marketplace.eclipse.org/content/contrast-eclipse."

[10] "Lint. https://developer.android.com/studio/write/lint.html."

[11] "Aside. https://www.owasp.org/index.php/OWASP_ASIDE_Project."

[12] J. Xie, B. Chu, H. R. Lipford, and J. T. Melton, "Aside: Ide support for web application security," in *Proceedings of the 27th Annual Computer Security Applications Conference*. ACM, 2011, pp. 267–276.

[13] "Codedx. http://codedx.com/ide-integration-helps-developers-adopt-application-security-testing-tools/."

[14] "Codeproanalytix. https://developers.google.com/java-dev-tools/codepro/doc/."

[15] "Cppcheclipse. https://marketplace.eclipse.org/content/cppcheclipse."

[16] "Checkmark cxsast. https://www.checkmarx.com/technology/static-code-analysis-sca/."

[17] "Esvd. https://marketplace.eclipse.org/content/early-security-vulnerability-detector-esvd."

[18] L. S. M. de Souza, "Early vulnerability detection for supporting secure programming," Master's thesis, Departamento de Informtica, Pontifcia Universidade Catlica do Rio de Janeiro, 2015. [Online]. Available: http://thecodemaster.net/wp-content/uploads/2015/06/early-vulnerability-detection-for-supporting-secure-programming.pdf

[19] "Findbugs. https://androidbycode.wordpress.com/2015/02/13/static-code-analysis-automation-using-findbugs-android-studio/."

[20] "Fortify. https://marketplace.eclipse.org/content/hpe-security-fortify-demand-plugin."

[21] "Fxcop. https://msdn.microsoft.com/en-us/library/bb429476(v=vs.80).aspx."

[22] "Goanna studio. https://marketplace.eclipse.org/content/goanna-studio-static-analysis-cc."

[23] "Klockwork. http://www.klocwork.com/products-services/klocwork/static-code-analysis."

[24] "Lapse+. https://code.google.com/p/lapse-plus/."

[25] P. M. Pérez, J. Filipiak, and J. M. Sierra, "Lapse+ static analysis security software: Vulnerabilities detection in java ee applications," in *Future Information Technology*. Springer, 2011, pp. 148–156.

[26] "Secureassist. https://www.cigital.com/resources/datasheets/secureassist-datasheet/."

[27] "Sensiolabsinsight. https://plugins.jetbrains.com/plugin/7589?pr=."

[28] "sonarlint. http://www.sonarlint.org/eclipse/index.html."

[29] "Ssv checker. https://marketplace.eclipse.org/content/ssvchecker."

[30] J. Dehlinger, Q. Feng, and L. Hu, "Ssvchecker: unifying static security vulnerability detection tools in an eclipse plug-in," in *Proceedings of the 2006 OOPSLA workshop on eclipse technology eXchange*. ACM, 2006, pp. 30–34.

[31] "Veracode. https://www.veracode.com/."

[32] "Findbugs. http://findbugs.sourceforge.net/."