CodeShield

Ganzheitliche Software Sicherheitsprüfung

Garage33

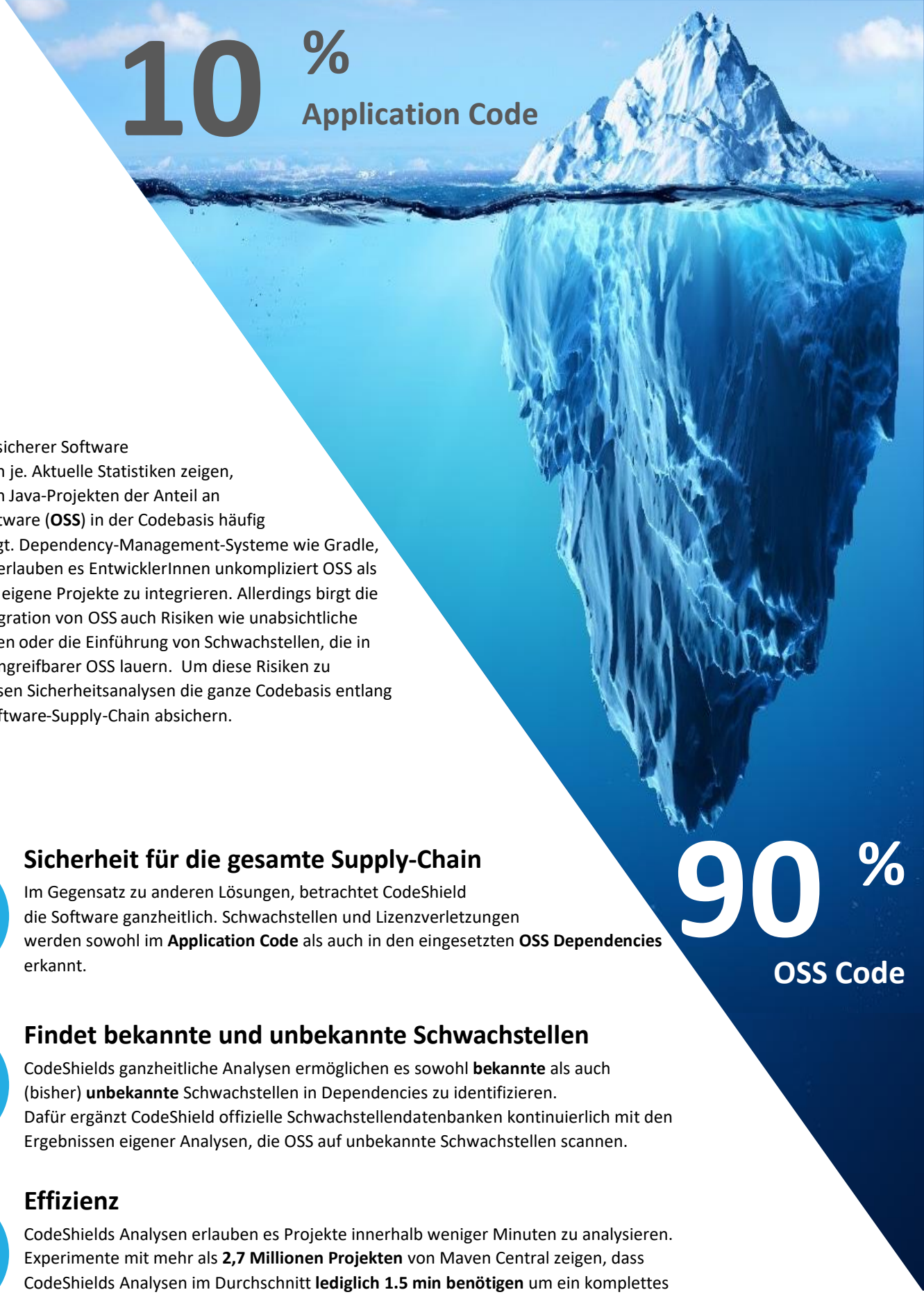
Technologiepark 8
33100 Paderborn

Dr. Johannes Späth
johannes.spaeth@codeshield.de

Manuel Benz, M.Sc.
manuel.benz@codeshield.de

Andreas Dann, M.Sc.
andreas.dann@codeshield.de

www.codeshield.de



Die Entwicklung sicherer Software ist schwerer denn je. Aktuelle Statistiken zeigen, dass in modernen Java-Projekten der Anteil an Open-Source Software (**OSS**) in der Codebasis häufig **bis zu 90%** beträgt. Dependency-Management-Systeme wie Gradle, Maven und SBT, erlauben es EntwicklerInnen unkompliziert OSS als **Dependencies** in eigene Projekte zu integrieren. Allerdings birgt die unachtsame Integration von OSS auch Risiken wie unabsichtliche Lizenzverletzungen oder die Einführung von Schwachstellen, die in veralteter oder angreifbarer OSS lauern. Um diese Risiken zu eliminieren, müssen Sicherheitsanalysen die ganze Codebasis entlang der gesamten Software-Supply-Chain absichern.

Sicherheit für die gesamte Supply-Chain

Im Gegensatz zu anderen Lösungen, betrachtet CodeShield die Software ganzheitlich. Schwachstellen und Lizenzverletzungen werden sowohl im **Application Code** als auch in den eingesetzten **OSS Dependencies** erkannt.

Findet bekannte und unbekannte Schwachstellen

CodeShields ganzheitliche Analysen ermöglichen es sowohl **bekannte** als auch (bisher) **unbekannte** Schwachstellen in Dependencies zu identifizieren.
Dafür ergänzt CodeShield offizielle Schwachstellendatenbanken kontinuierlich mit den Ergebnissen eigener Analysen, die OSS auf unbekannte Schwachstellen scannen.

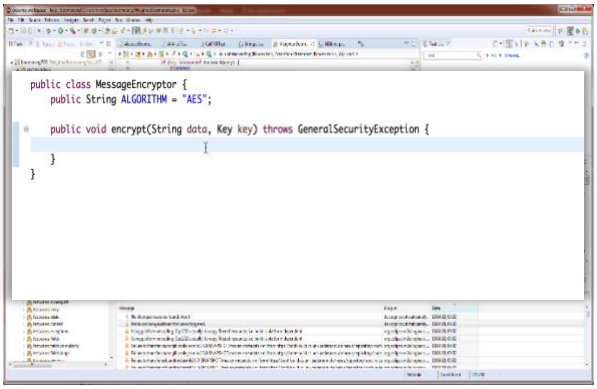
Effizienz

CodeShields Analysen erlauben es Projekte innerhalb weniger Minuten zu analysieren. Experimente mit mehr als **2,7 Millionen Projekten** von Maven Central zeigen, dass CodeShields Analysen im Durchschnitt **lediglich 1.5 min benötigen** um ein komplettes Softwareprojekt **vollständig** auf Schwachstellen zu analysieren. [1]

Präzision

CodeShields Analysen produzieren kaum Falschmeldungen.
So lag der Anteil an **Falsch-Positiven** bei der Analyse von **10.000 Android Applikationen** bzgl. Der fehlerhaften Nutzung von Kryptographie bei lediglich **8 Prozent.** [2]

Application Code Analyse



Die Application Code Analysen von CodeShield erlauben es bereits während der Entwicklung Schwachstellen aufzudecken.

CodeShield lässt sich direkt in gängige IDEs und CI-Systeme integrieren und gibt EntwicklerInnen unmittelbares Feedback - ähnlich einer Rechtschreibprüfung.

Falsche Nutzung kryptographischer Primitiven

CodeShields Analysen können die falsche Nutzung kryptographischer Primitiven effizient aufdecken und die korrekte Verwendung von Verschlüsselungsalgorithmen, Erzeugung von Schlüsseln, Einsatz von Verschlüsselungsmodi, Padding, etc., aufzeigen.

Security Analysen: OWASP 10 & SANS 25

Zur Aufdeckung häufiger **Programmierfehler und damit einhergehender Sicherheitslücken** (SQL-Injection, XSS, etc.) erkennt CodeShield die in der OWASP Top 10 und SANS 25 spezifizierten kritischsten Typen von Sicherheitslücken.

Java Bytecodeanalyse

CodeShield arbeitet direkt auf Java **Bytecode**. So können Sicherheitslücken auch in dem von **Frameworks** generiertem Bytecode aufgedeckt werden.

Demo

Eine erste Version der Application Code Analyse ist in CogniCrypt integriert. CogniCrypt ist ein Eclipse Plugin, das kryptographische Fehlbenutzungen erkennt. Eine aktuelle Version ist unter https://www.eclipse.org/cognicrypt/ verfügbar.

Presseartikel

https://www.heise.de/developer/artikel/CogniCrypt-Kryptografie-richtig-nutzen-4211551.html

https://www.golem.de/news/software-entwickler-krypto-fehler-vermeiden-mit-cognicrypt-1905-141609.html

OSS Dependency Analyse

Die Absicherung von Dependencies stellt ganz neue Herausforderungen an die sichere Softwareentwicklung. EntwicklerInnen müssen schon bei der Einbindung von OSS-Dependencies überprüfen, ob diese von Sicherheitslücken betroffen sind, die bereits in **bekannten** Datenbanken

(NVD, Exploit DB) veröffentlicht wurden, oder bestehende Lizenzen verletzen. CodeShields Dependency Scanner erkennt die eingesetzten Dependencies und kann vor bereits veröffentlichten Schwachstellen und Lizenzverletzungen warnen.

Identifiziert bekannte und unbekannte Schwachstellen

CodeShield analysiert die eingesetzten Dependencies mit seinen Application Code Analysen. Dies erlaubt es auch bislang unbekannte Schwachstellen zu identifizieren, zusätzlich zu den bereits bekannten Schwachstellen für diese Dependencies.

Rebundling, Repackaging, Recompilation

CodeShield verwendet zur Erkennung eingesetzter Dependencies nicht nur ausschließlich Metadaten (GAV), sondern analysiert den Bytecode und kann so zuverlässig modifizierte Dependencies und Codeduplicate (wieder-)erkennen. [3]
So lassen sich auch Schwachstellen in den auf Maven Central weit verbreiteten Service-Bundles und in OSS die vom Unternehmen modifiziert wurde identifizieren.

Risikobewertung

Zur Bewertung der Auswirkungen von Schwachstelle in eingebundenen Dependency bietet der Dependency Scanner eine Erreichbarkeitsanalyse an. Diese aufzeigt auf, ob und wie der schwachstellenbehaftete Code genutzt wird. So kann das Risiko für eigene Projekte bewertet und ggf. unnötige Neuauslieferungen und Updates beim Nutzer verhindert werden.

Lizenzverletzungen

CodeShields **effiziente** Erkennung eingebundener Dependencies erlaubt es eine Übersicht der verwendeten Lizenzen anzufertigen, um so Lizenzverletzungen, die durch den Einsatz mehrere OSS-Dependencies mit inkompatiblen Lizenzen entstehen, frühzeitig zu erkennen.

Publikationen

[1] *Synchronized Pushdown Systems for Pointer and Data-Flow Analysis*; Universität Paderborn 2019; Johannes Späth

[2] *CrySL: An Extensible Approach to Validating the Correct Usage of Cryptographic APIs*; ECOOP 2018; Stefan Krüger, Johannes Späth, Karim Ali, Eric Bodden, Mira Mezini

[3] *SootDiff – Comparing Bytecode across Different Java Compiler*; SOAP 2019; Andreas Dann, Ben Hermann, Eric Bodden