
SUPPLY CHAIN SECURITY

Sicherheitslücken in eigener und fremder Software identifizieren

Dr.-Ing. Steven Arzt

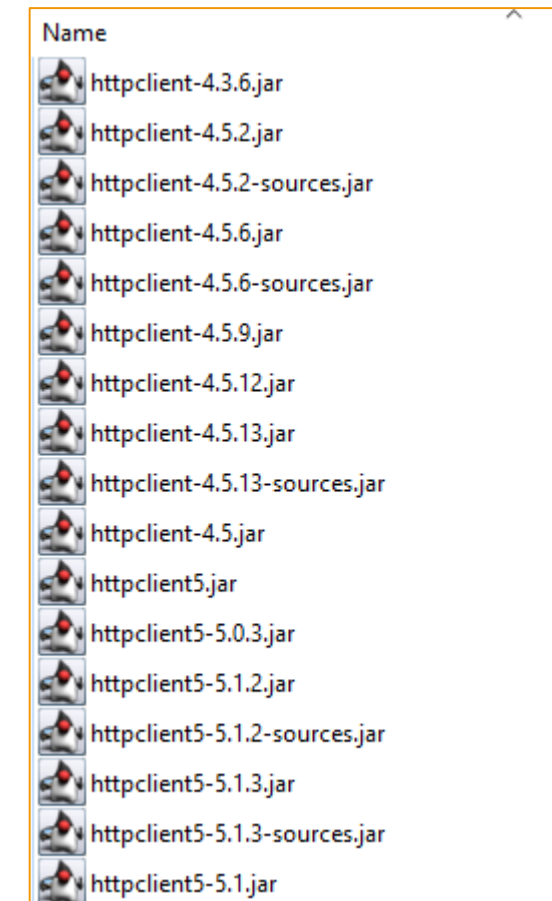


ATHENE
Nationales Forschungszentrum
für angewandte Cybersicherheit



ABHÄNGIGKEITEN ÜBERWACHEN

- SBOM: Software Bill of Materials
 - Was wird wo verwendet in welcher Version?
- Schwachstellenmanagement
 - Welche Versionen sind verwundbar?
- Risikomanagement
 - Auswirkungen der Schwachstelle
- Patchmanagement



WIE LANGE GAB ES CVE-2021-44832 ?

	2.17.0	2 vulnerabilities	Central	965	Dec 18, 2021
2.16.x	2.16.0	3 vulnerabilities	Central	988	Dec 13, 2021
2.15.x	2.15.0	5 vulnerabilities	Central	1,179	Dec 10, 2021
2.14.x	2.14.1	8 vulnerabilities	Central	1,092	Mar 12, 2021
	2.14.0	8 vulnerabilities	Central	1,079	Nov 10, 2020
2.13.x	2.13.3	8 vulnerabilities	Central	1,342	May 14, 2020
	2.13.2	8 vulnerabilities	Central	874	Apr 21, 2020
	2.13.1	9 vulnerabilities	Central	343	Feb 29, 2020
	2.13.0	9 vulnerabilities	Central	403	Dec 12, 2019
2.12.x	2.12.4	6 vulnerabilities	Central	117	Dec 29, 2021
	2.12.3	8 vulnerabilities	Central	74	Dec 22, 2021
	2.12.2	9 vulnerabilities	Central	33	Dec 14, 2021
	2.12.1	9 vulnerabilities	Central	1,485	Aug 09, 2019
	2.12.0	9 vulnerabilities	Central	264	Jun 29, 2019
2.11.x	2.11.2	9 vulnerabilities	Central	547	Feb 09, 2019
	2.11.1	9 vulnerabilities	Central	1,424	Jul 23, 2018
	2.11.0	9 vulnerabilities	Central	1,025	Mar 11, 2018
2.10.x	2.10.0	9 vulnerabilities	Central	704	Nov 19, 2017

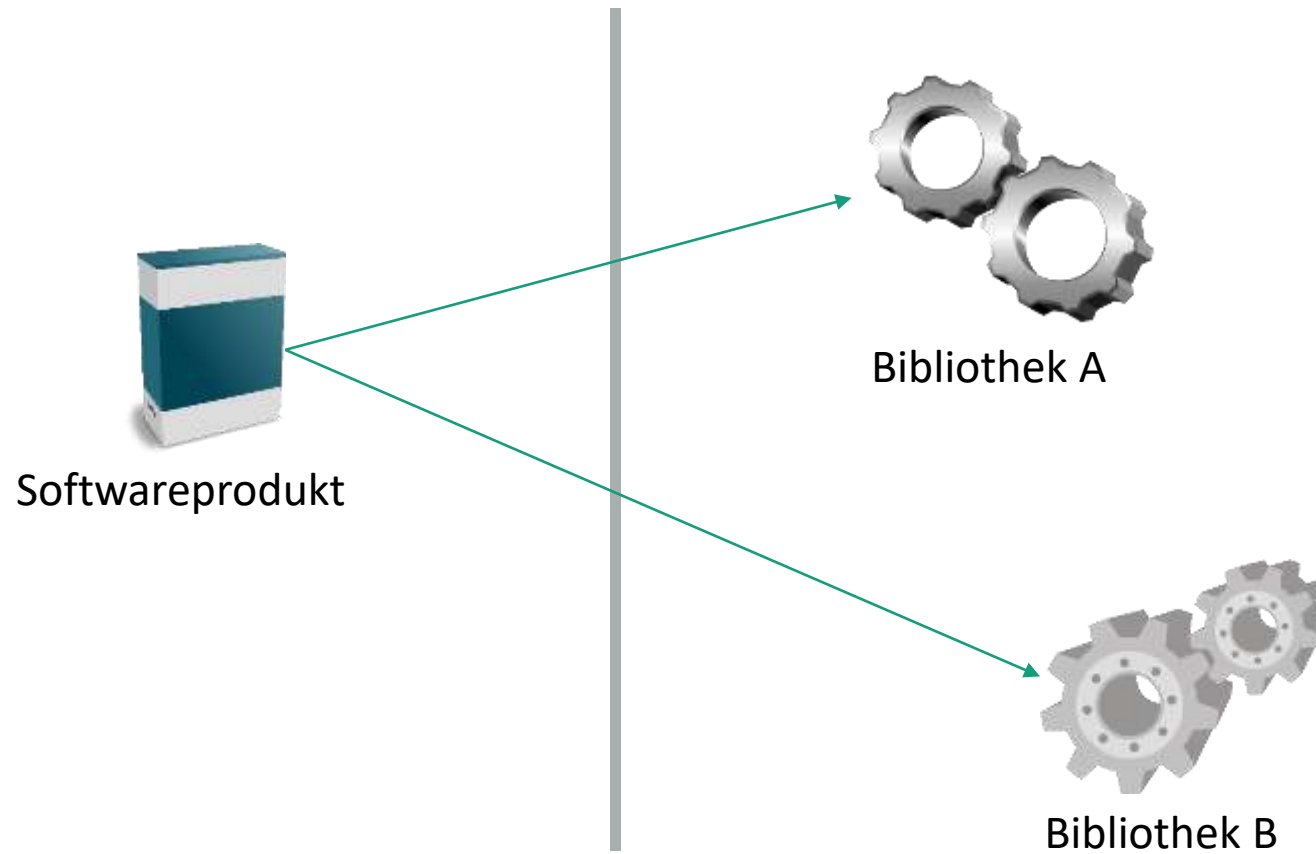
2.9.x	2.9.1	9 vulnerabilities	Central	992	Sep 18, 2017
	2.9.0	9 vulnerabilities	Central	394	Aug 26, 2017
2.8.x	2.8.2	9 vulnerabilities	Central	1,197	Apr 02, 2017
	2.8.1	10 vulnerabilities	Central	387	Feb 27, 2017
	2.8	10 vulnerabilities	Central	114	Jan 22, 2017
2.7.x	2.7	10 vulnerabilities	Central	1,021	Oct 02, 2016
2.6.x	2.6.2	10 vulnerabilities	Central	836	Jul 06, 2016
	2.6.1	10 vulnerabilities	Central	248	Jun 06, 2016
	2.6	10 vulnerabilities	Central	167	May 25, 2016
2.5.x	2.5	10 vulnerabilities	Central	620	Dec 07, 2015
2.4.x	2.4.1	10 vulnerabilities	Central	128	Oct 09, 2015
	2.4	10 vulnerabilities	Central	35	Sep 20, 2015
2.3.x	2.3.2	7 vulnerabilities	Central	10	Dec 29, 2021
	2.3.1	9 vulnerabilities	Central	11	Dec 22, 2021
	2.3	10 vulnerabilities	Central	273	May 10, 2015
2.2.x	2.2	10 vulnerabilities	Central	179	Feb 22, 2015
2.1.x	2.1	10 vulnerabilities	Central	266	Oct 20, 2014
	2.0.2	10 vulnerabilities	Central	64	Aug 17, 2014
	2.0.1	10 vulnerabilities	Central	34	Jul 30, 2014
	2.0	10 vulnerabilities	Central	43	Jul 12, 2014

WER PRÜFT DIE BIBLIOTHEKEN?

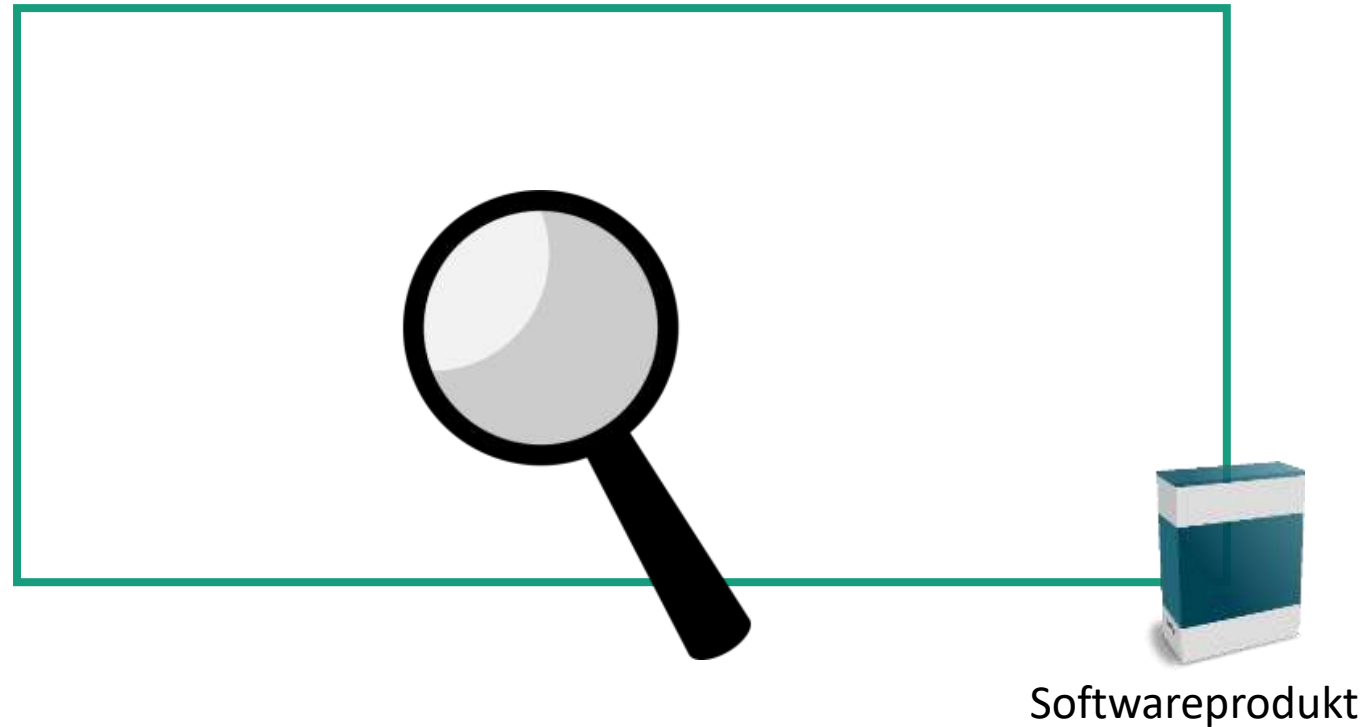
- Open Source = Sicher?
 - Code prüfbar heißt nicht Code wird geprüft!
 - Viele Projekte haben nur wenige Beitragende
 - Sicherheitskonzepte stark unterschiedlich
- Probleme in großen Projekten
 - Log4jshell
 - Spring4shell
 - Debian OpenSSH



ERWARTUNGEN ENTWICKLER VS. ANWENDER (1)



ERWARTUNGEN ENTWICKLER VS. ANWENDER (2)



ERKENNEN VON SICHERHEITSLÜCKEN



Pentest



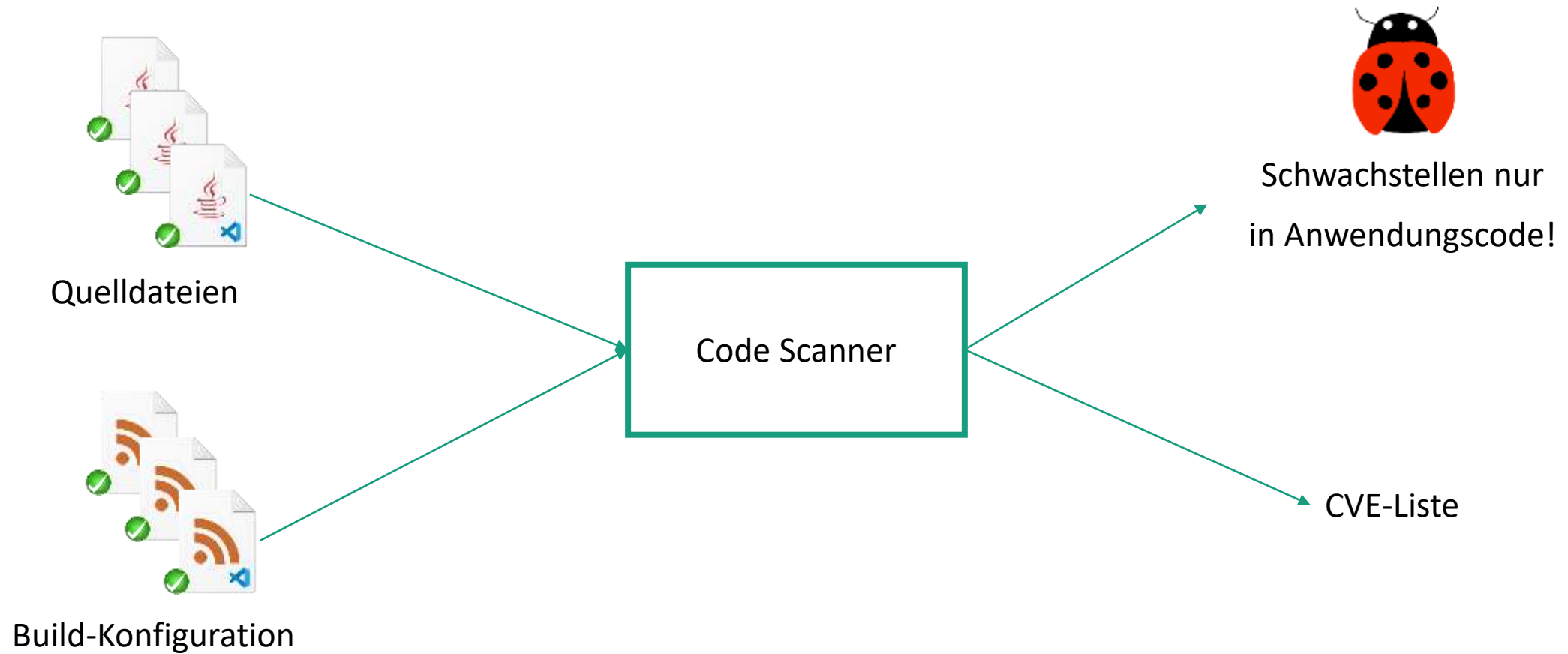
Code Review



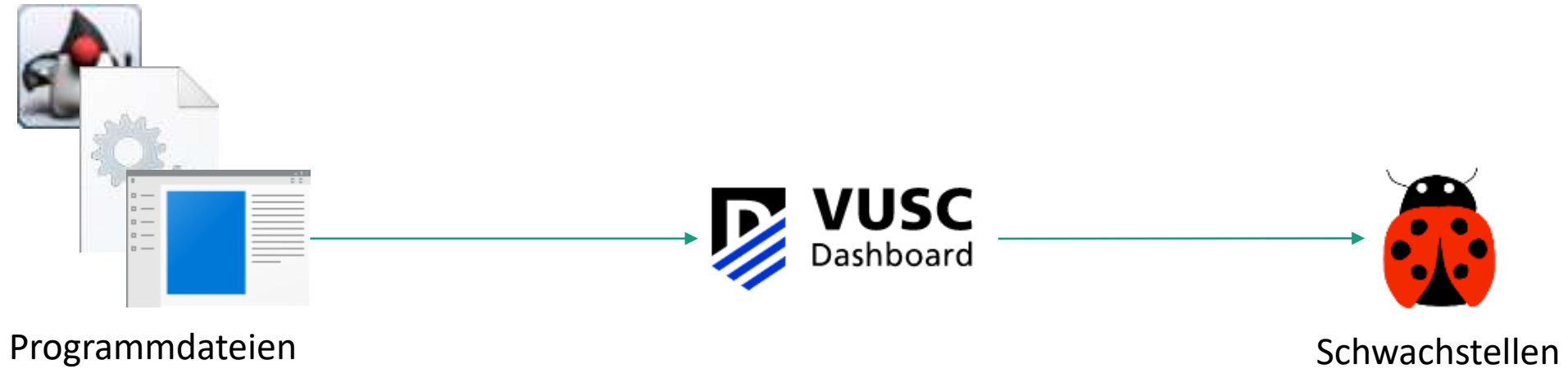
Manuelle Verfahren

Automatisierte Verfahren

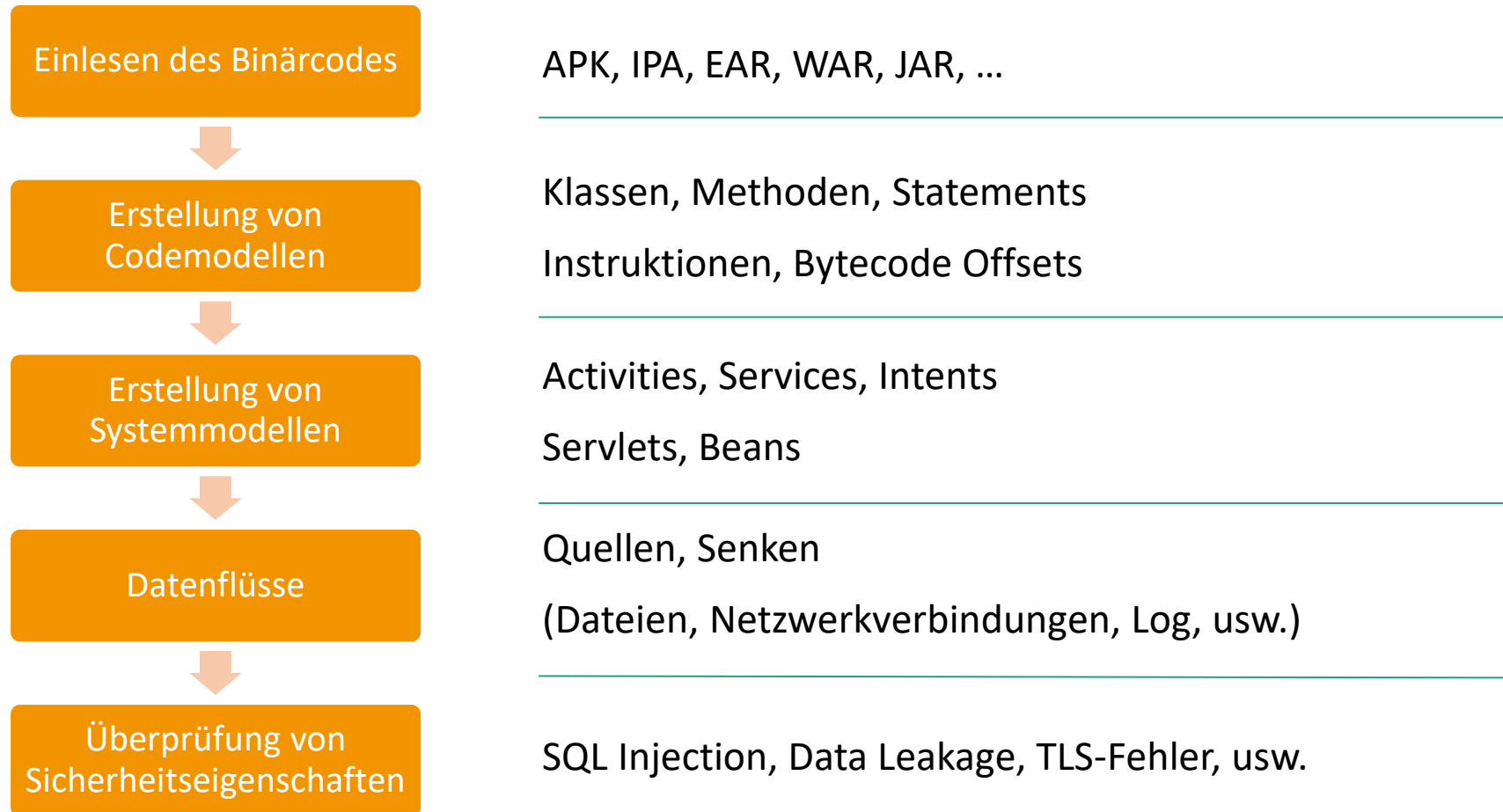
QUELLCODE SCANNEN



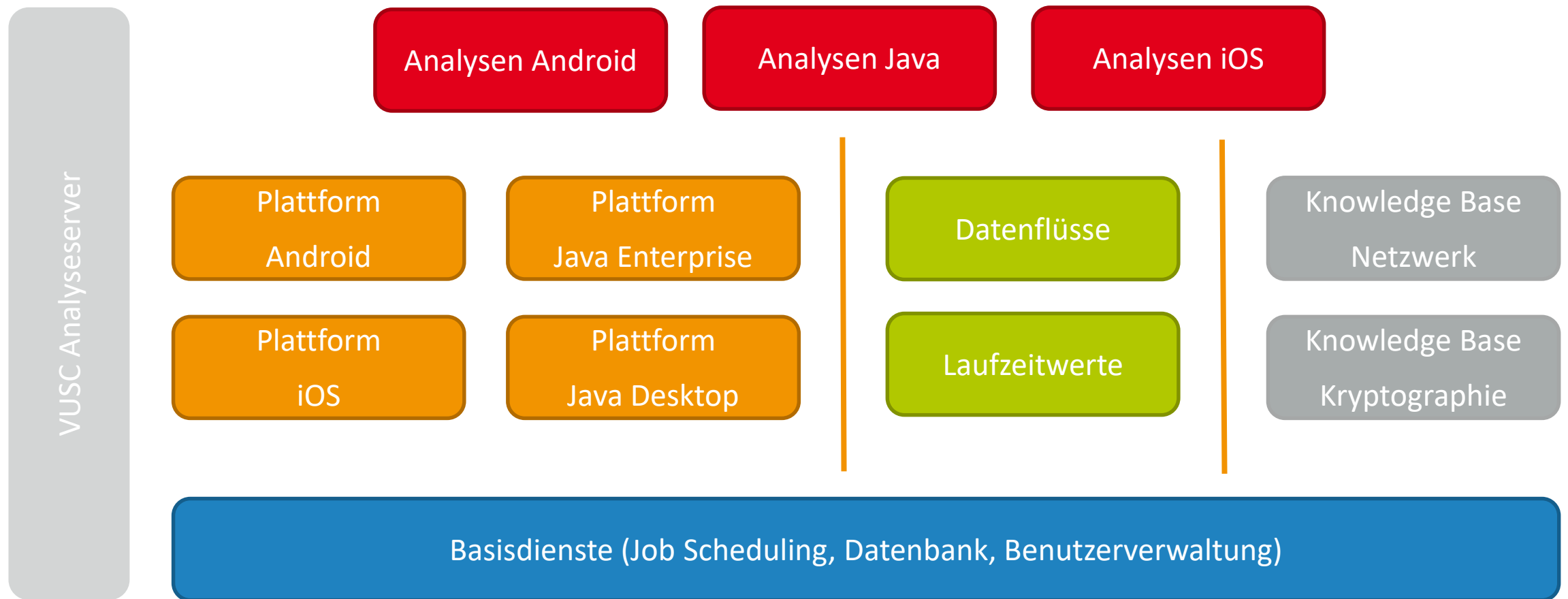
BINÄRCODE SCANNEN



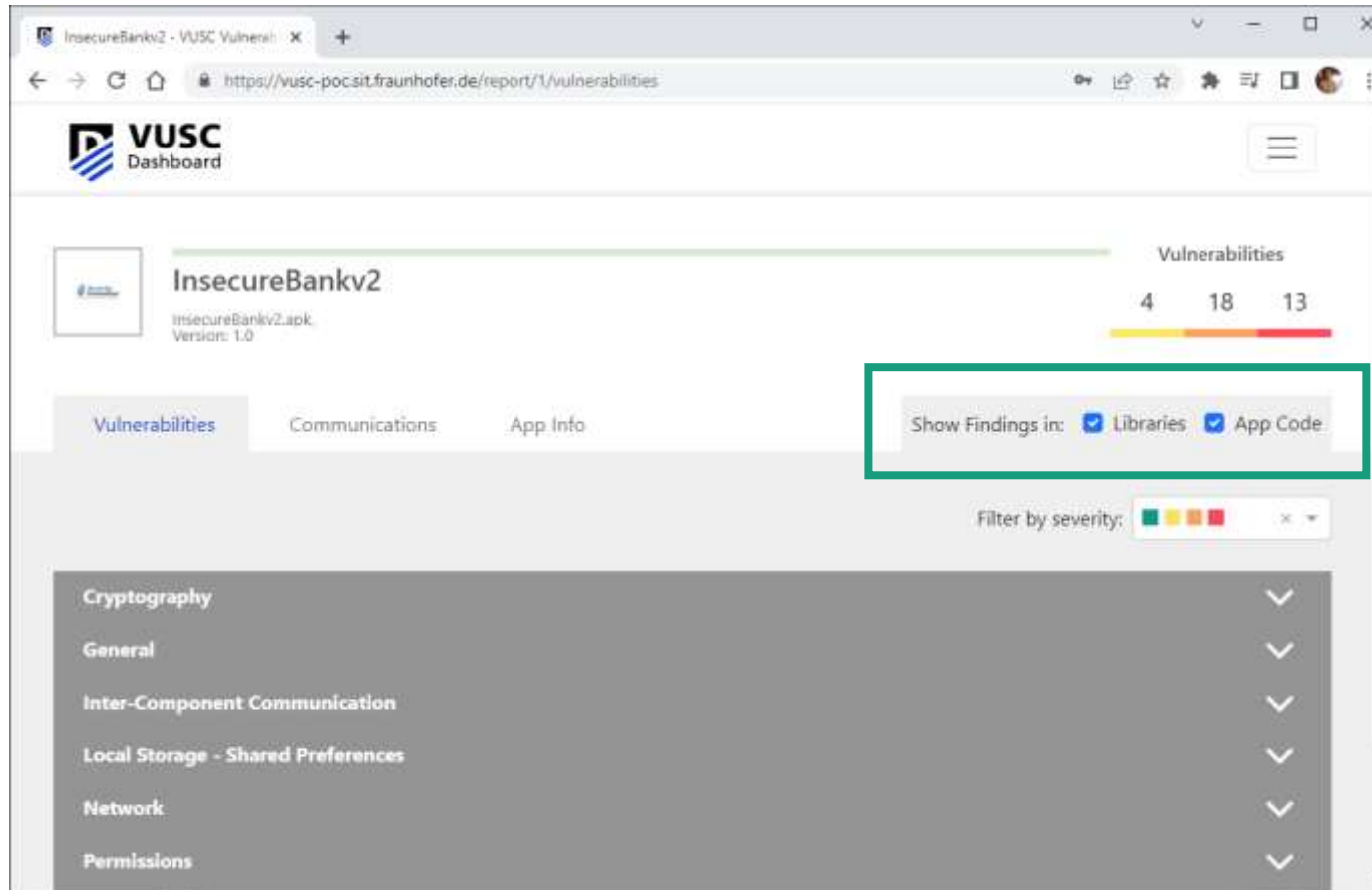
FUNKTIONSPRINZIP VUSC



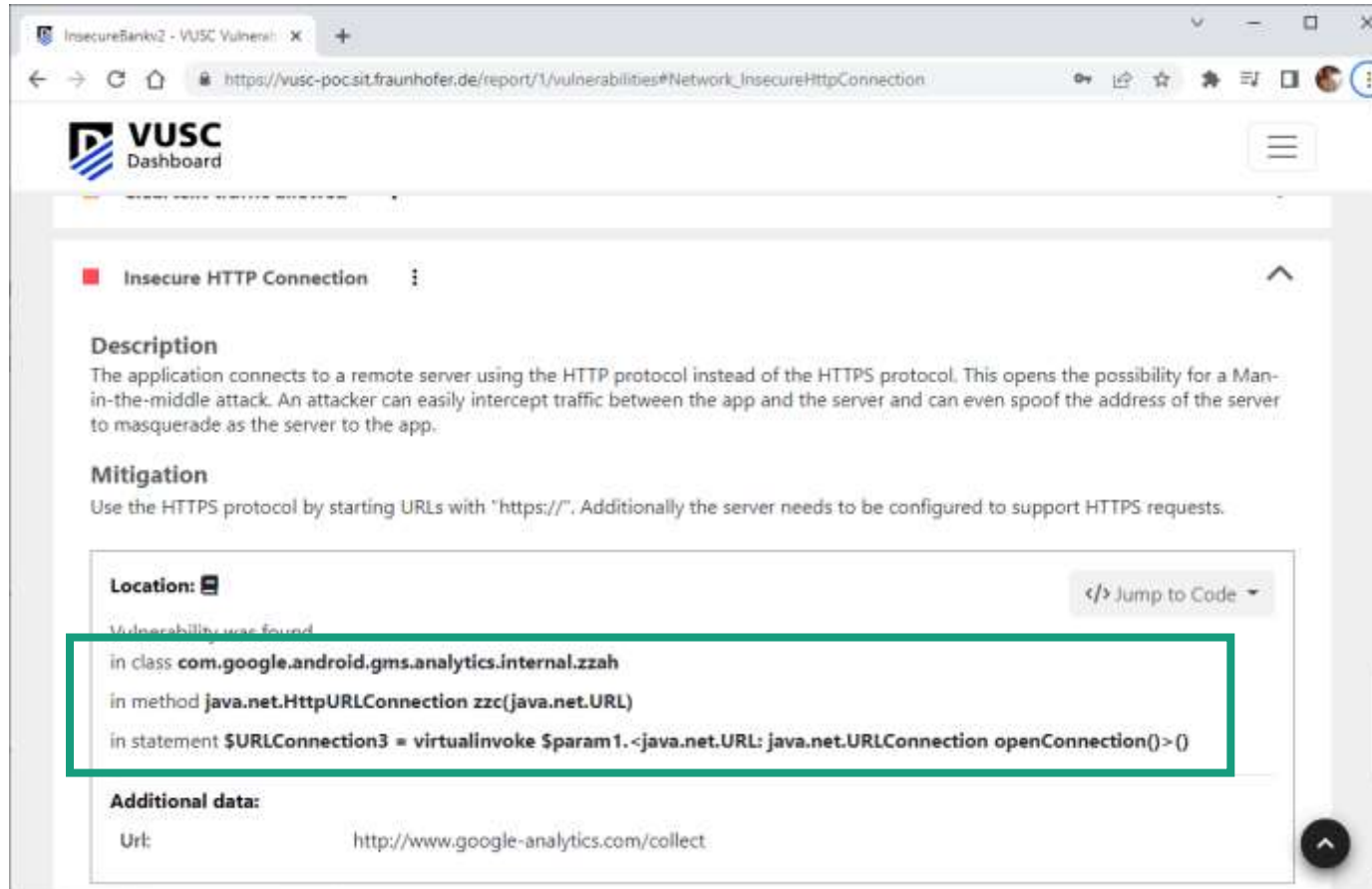
SYSTEMARCHITEKTUR VUSC



VUSC FINDET SCHWACHSTELLEN



SCHWACHSTELLEN IN BIBLIOTHEKEN



The screenshot shows a web browser window displaying the VUSC Dashboard. The URL in the address bar is https://vusc-pocsit.fraunhofer.de/report/1/vulnerabilities#Network_InsecureHttpConnection. The dashboard header includes the VUSC logo and a menu icon. The main content area features a red square icon and the title "Insecure HTTP Connection". Below this, a "Description" section explains that the application uses HTTP instead of HTTPS, which is vulnerable to man-in-the-middle attacks. A "Mitigation" section advises using HTTPS and starting URLs with "https://". The "Location" section, highlighted with a green box, shows the vulnerability was found in the class `com.google.android.gms.analytics.internal.zzah`, in the method `java.net.HttpURLConnection zzc(java.net.URL)`, and in the statement `$URLConnection3 = virtualinvoke $param1.<java.net.URL: java.net.URLConnection openConnection()->()`. A button labeled "</> Jump to Code" is also present. The "Additional data" section shows the URL `http://www.google-analytics.com/collect`.

VUSC Dashboard

Insecure HTTP Connection

Description
The application connects to a remote server using the HTTP protocol instead of the HTTPS protocol. This opens the possibility for a Man-in-the-middle attack. An attacker can easily intercept traffic between the app and the server and can even spoof the address of the server to masquerade as the server to the app.

Mitigation
Use the HTTPS protocol by starting URLs with "https://". Additionally the server needs to be configured to support HTTPS requests.

Location:  </> Jump to Code

Vulnerability was found

- in class `com.google.android.gms.analytics.internal.zzah`
- in method `java.net.HttpURLConnection zzc(java.net.URL)`
- in statement `$URLConnection3 = virtualinvoke $param1.<java.net.URL: java.net.URLConnection openConnection()->()`

Additional data:

Url: `http://www.google-analytics.com/collect`

SBOM: ENTHALTENE BIBLIOTHEKEN

webgoat-server - VUSC Vulnerabilities

https://vusc-pocsit.fraunhofer.de/report/3/app-info/used-libraries

VUSC Dashboard

webgoat-server
webgoat-server-8.2.1.jar
Version: 8.2.1

Vulnerabilities: 4 High, 0 Medium, 5 Low

Vulnerabilities | Communications | **App Info**

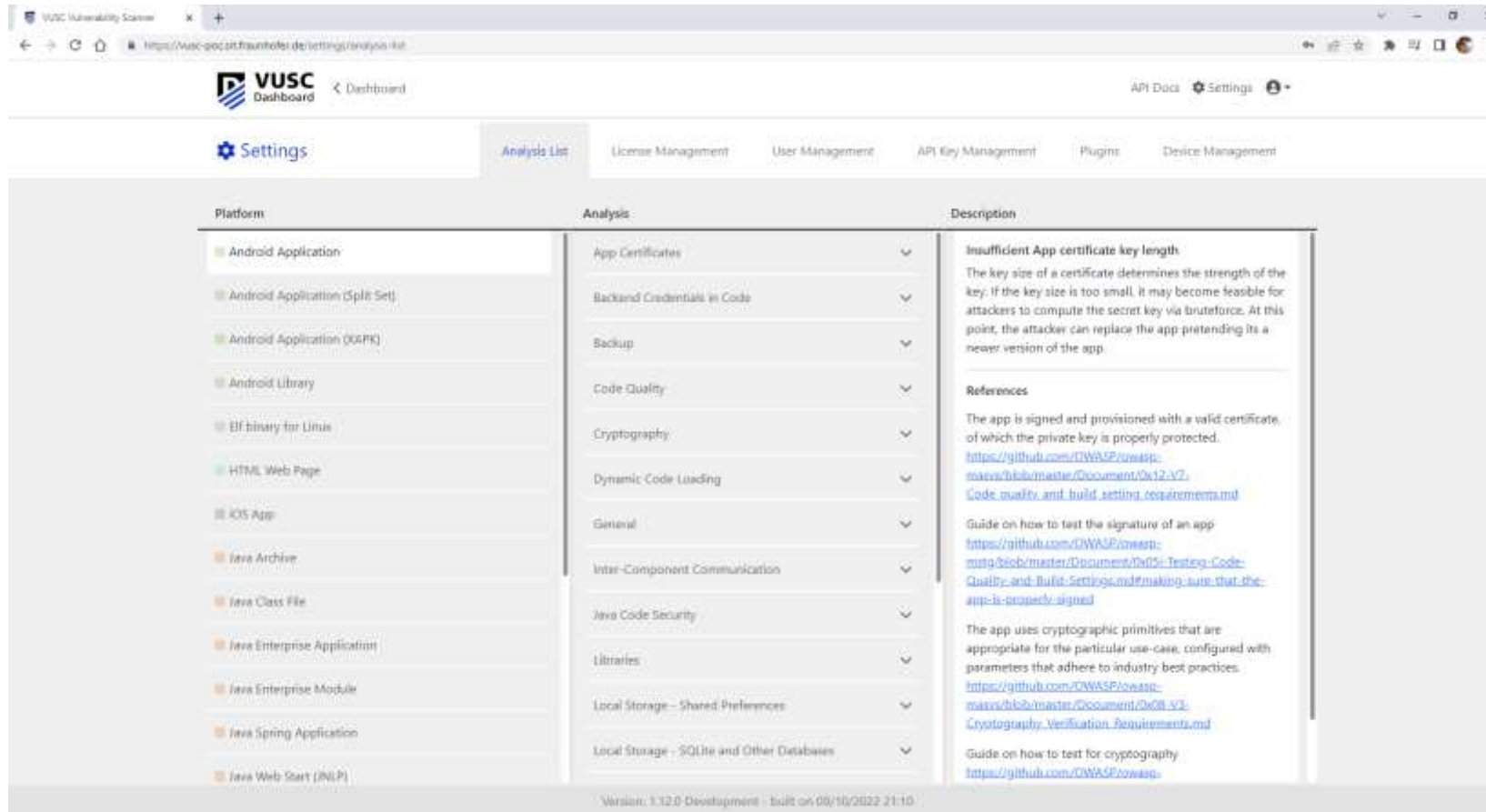
Search

Libraries
385x

Permissions
0x

Name	Version	Maven - Artifact Id	Maven - Group Id
me.qmx.jitescript:jitescript	0.4.1	jitescript	me.qmx.jitescript
com.github.jnr:jnr-ffi	2.2.0	jnr-ffi	com.github.jnr

ERKENNUNG VON ÜBER 380 SCHWACHSTELLEN



Platform	Analysis	Description
Android Application	App Certificates	Insufficient App certificate key length The key size of a certificate determines the strength of the key. If the key size is too small, it may become feasible for attackers to compute the secret key via brute force. At this point, the attacker can replace the app pretending its a newer version of the app. References The app is signed and provisioned with a valid certificate, of which the private key is properly protected. https://github.com/DWAP/owasp-mustgob/master/Document/OWASP-V7-Code-quality-and-build-setting-requirements.md Guide on how to test the signature of an app https://github.com/DWAP/owasp-mustgob/master/Document/OWASP-V7-Code-quality-and-build-setting-requirements.md#making-sure-that-the-app-is-properly-signed The app uses cryptographic primitives that are appropriate for the particular use-case, configured with parameters that adhere to industry best practices. https://github.com/DWAP/owasp-mustgob/master/Document/OWASP-V7-Cryptography-Verification-Requirements.md Guide on how to test for cryptography https://github.com/DWAP/owasp-mustgob/master/Document/OWASP-V7-Cryptography-Verification-Requirements.md
Android Application (Split Set)	Backend Credentials in Code	
Android Application (OATK)	Backup	
Android Library	Code Quality	
Elf binary for Linux	Cryptography	
HTML Web Page	Dynamic Code Loading	
iOS App	General	
Java Archive	Inter-Component Communication	
Java Class File	Java Code Security	
Java Enterprise Application	Libraries	
Java Enterprise Module	Local Storage - Shared Preferences	
Java Spring Application	Local Storage - SQLite and Other Databases	
Java Web Start (JNLP)		

Version: 1.12.0 Development - built on 08/10/2022 21:10

Vielen Dank für Ihre Aufmerksamkeit!

Dr.-Ing. Steven Arzt

Abteilungsleiter

Secure Software Engineering

Fraunhofer-Institut für Sichere Informationstechnologie



Rheinstraße 75 | 64295 Darmstadt | Germany

Telefon +49 6151 869-336 | Fax +49 6151 869-224

steven.arzt@sit.fraunhofer.de | www.sit.fraunhofer.de