Im Auftrag von:

Bundesamt für Sicherheit in der Informationstechnik

mgm security partners

Deutschland Digital•Sicher•BSI•

Ergebnisse der Studie zur statischen Codeanalyse ausgewählter Opensource Software

Die Studie wurde im Auftrag des BSI erstellt.

Videokonferenzen

Ergebnisbericht
Security Quellcodeanalyse und Penetration Test

Projekt 486
Codeanalyse für Opensource Software (CAOS)

Analyseergebnisse Arbeitspaket 3 „Videokonferenzsysteme“

25.04.2022, Version 1.0, Status: Final

mgm security partners GmbH
Taunusstraße 23
80807 München
Tel.: +49/89/358680-880
E-Mail: info@mgm-sp.com
http://www.mgm-sp.com

A. Zusammenfassung und Bewertung

Sprungtabelle

Auf den folgenden Seiten sind die Ergebnisse der Sicherheitsanalyse zusammengefasst.

Alle Verweise in diesem Dokument sind aktive Links, d. h. können per Mausklick direkt angesprungen werden.

A 1 Inhaltsverzeichnis

A 2 Management Summary

Der vorliegende Bericht fasst die Ergebnisse der Untersuchung folgender zwei Open-Source Video-Konferenzsysteme zusammen (Details s. A4.1):

- BigBlueButton
- Jitsi

Beide Anwendungen wurden einer (jeweils manuell und semiautomatisch durchgeführten) statischen Quellcode-Analyse mit begleitender dynamischer Analyse unterzogen. Die Untersuchung erfolgte nach dem Whitebox-Verfahren, bei welchem die Tester sowohl Zugriff auf den Code, als auch auf lauffähige dedizierte Instanzen hatten.

Die Analysen fanden im Zeitraum von 01.02.2022 bis 25.04.2022 in den Räumlichkeiten des Auftragnehmers statt und wurden über das interne Netzwerk durchgeführt.

Ergebnis BigBlueButton

Die Anwendung weist zwei als

hoch eingestufte Sicherheitslücken

auf, mit denen ein Angreifer gezielt Benutzer und die Anwendung kompromittieren kann. Es wurden dafür zwei CVEs (Common Vulnerability Enumeration) registriert (CVE-2022-26497 und CVE-2022-27238), sowie nach den Regeln der „Responsible Disclosure“ Kontakt mit dem Entwicklungsteam aufgenommen. Dieser verlief sehr positiv und freundlich. Die Entwickler waren an unseren Erkenntnissen interessiert und bekundeten auch Interesse an allen „weiteren Beobachtungen“. Eine der beiden Schwachstellen wurde auch sehr kurzfristig behoben (s. C1.2).

Bei beiden CVEs handelt es sich um Stored Cross-Site-Scripting Schwachstellen, welche ein sehr hohes Gefährdungspotential besitzen (s. E1.1.1 und F1.1.1).

Die Anwendung verwendet zudem 75 Abhängigkeiten mit Known Vulnerabilities, welche von einschlägigen Datenbanken mit [kritisch] oder [hoch] eingestuft werden (s. die SCA-Übersicht in A3.1.1).

Weitere, deutlich weniger sicherheitskritische Auffälligkeiten, welche nicht von automatischen SAST- (Static Application Security Testing) oder SCA- (Software Composition Analysis) Analysen aufgefunden wurden, sind in den Kapiteln E1 und F1 dokumentiert. Unter anderem ist es möglich, per Brute-Force valide Raumnamen einer Instanz herauszufinden (s. F1.4.1, als [niedrig] eingestuft).

Die gesamtheitliche Code-Qualität und auch die Herangehenweisen an die Korrektur von Schwachstellen, lassen an einigen Stellen zu wünschen übrig bzw. werfen sie Fragen auf (s. z.B. unsere CVE-Korrektur-Analysen in Kapitel C1). Dies zeigt sich auch in der recht hohen Anzahl an SAST- und SCA-Findings (s. A3.1.1). Im SAST-Bereich existiert eine Vielzahl an Findings mit dem Gefährdungspotential [mittel] und darunter.

Einige, als „schlechte Praxis“ bewertete Findings, wirken auch mehr wie übliche Programmierfehler, welche von Werkzeugen, wie Findbugs etc. aufgefunden werden

würden. Dies deutet darauf hin, dass im Projekt vermutlich keine regelmäßigen, werkzeuggestützten Code-Qualiätsprüfungen oder manuelle Code-Reviews durchgeführt werden.

Nichtsdestotrotz darf man im Vergleich mit Jitsi nicht vergessen, dass BigBlueButton über einen deutlich höheren Funktionsumfang und auch eine höhere Codezeilen-Anzahl verfügt.

Ergebnis Jitsi

Die Anwendung weist keine schwerwiegenden Sicherheitslücken auf.

Die Anwendung verwendet 31 Abhängigkeiten mit Known Vulnerabilities, welche von einschlägigen Datenbanken mit [kritisch] oder [hoch] eingestuft werden (s. die SCA-Übersicht in A3.1.1).

Zwei als [info] eingestufte Findings, welche nicht von automatischen SAST- (Static ApplicationSecurity Testing) oder SCA- (Software Composition Analysis) Analysen aufgefunden wurden, sind in Kapitel E2 dokumentiert.

Auch bei diesem Projekt lässt die gesamtheitliche Code-Qualität an einigen Stellen zu wünschen übrig. So haben wir beispielsweise auch hier Code-Stellen bewertet, welche eher klare Programmierfehler („Code Quality“) darstellen und von Werkzeugen, wie Findbugs etc. aufgefunden werden würden. Dies deutet ebenfalls darauf hin, dass vermutlich keine regelmäßigen, werkzeuggestützten Code-Qualitäsprüfungen oder manuelle Code-Reviews durchgeführt werden.

Auch für Jitsi wurde eine verhältnismäßig hohe Anzahl an SAST- und SCA-Findings gemeldet (s. A3.1.2).

A 3 Übersicht aller Findings

A 3.1 Toolbasierte Findings

Dem in B1.2 beschriebenen Vorgehen zur Bestimmung des Gefährdungspotentials folgend, werden in diesem Kapitel alle Ergebnisse automatischer Toolanalysen zusammengefasst dargestellt.

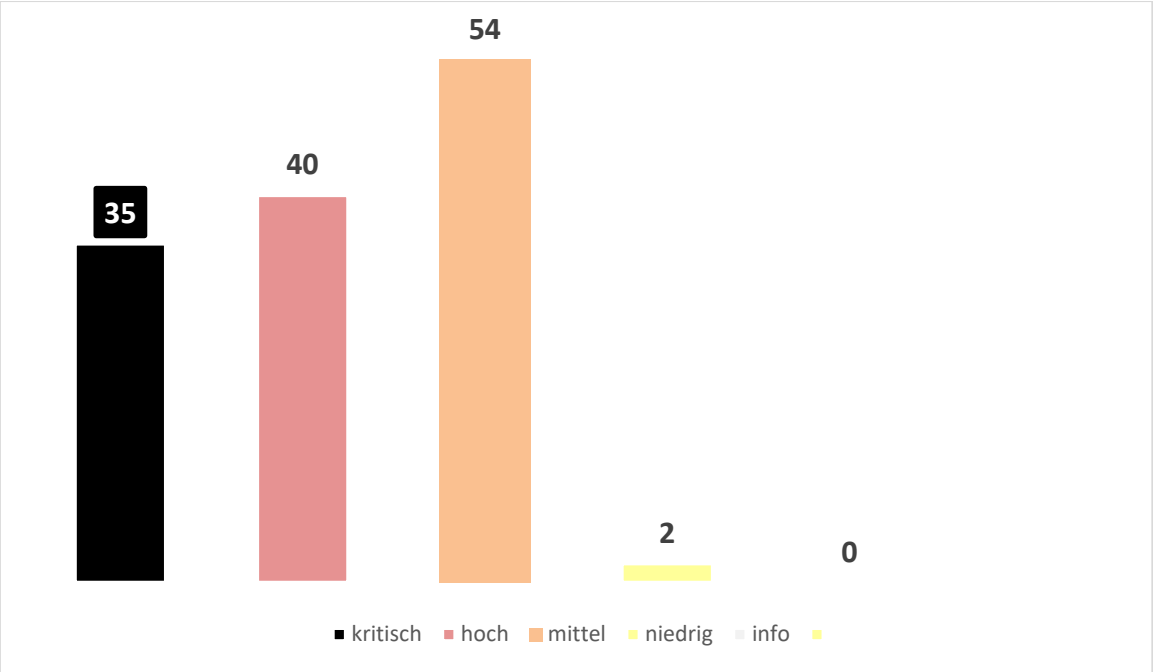
A 3.1.1 BigBlueButton

SAST-Analysen

Auf Basis der durchgeführten semiautomatischen SAST-Analysen (s. Abschnitte D1.1, D2.1, D3.1, D4.1) ergeben sich für das Videokonferenz-System BigBlueButton, nach Abbildung auf unser Gefährdungspotential, folgende Mengen an Findings:

SCA-Analyse

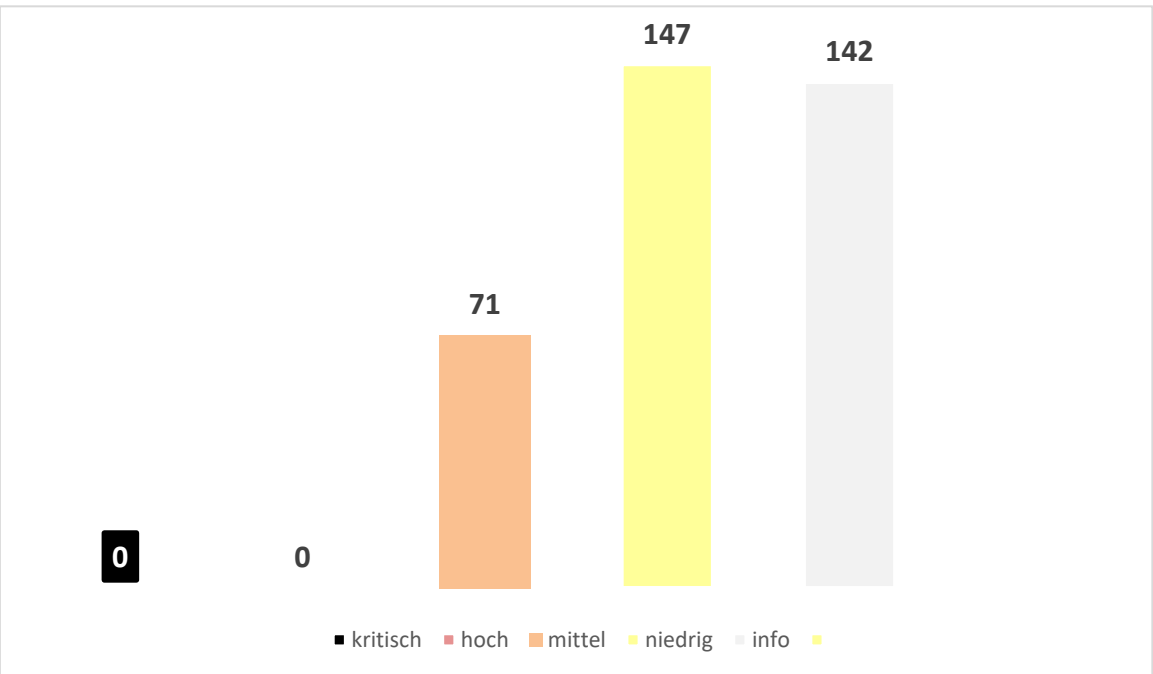
Auf Basis der automatischen SCA-Analyse (s. Abschnitt D6.1) ergeben sich für das Videokonferenz-System BigBlueButton, nach Abbildung auf unser Gefährdungspotential, folgende Mengen an Findings:



A 3.1.2 Jitsi

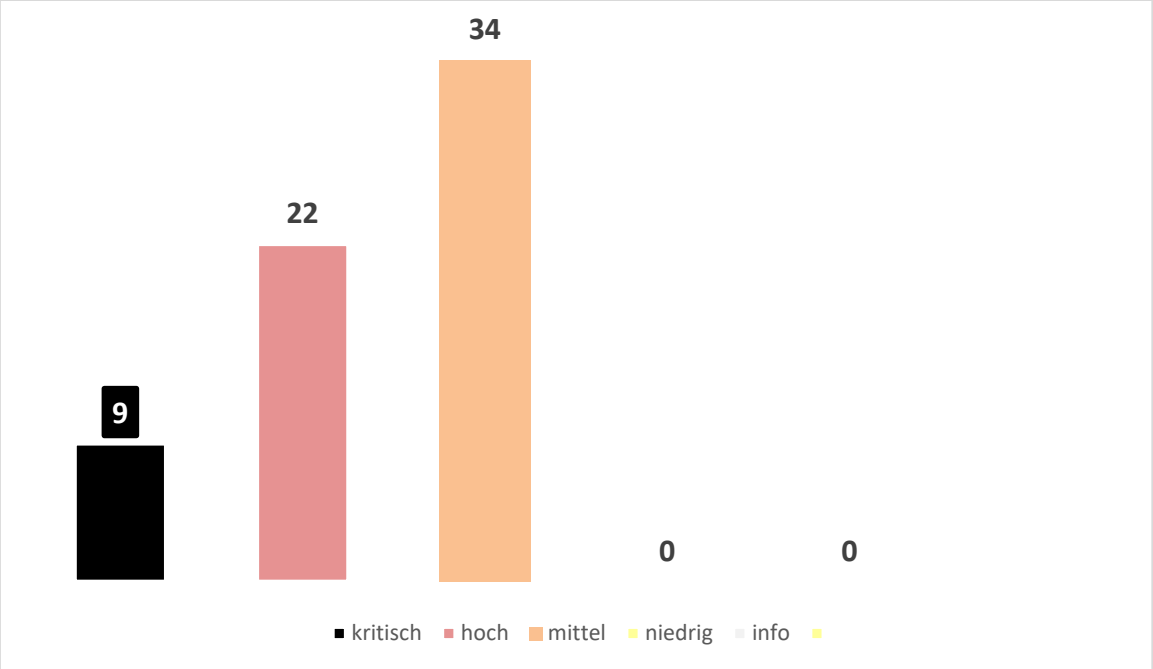
SAST-Analysen

Auf Basis der durchgeführten semiautomatischen SAST-Analysen (s. Abschnitte D1.2, D2.2, D3.2, D4.2) ergeben sich für das Videokonferenz-System Jitsi, nach Abbildung auf unser Gefährdungspotential, folgende Mengen an Findings:



SCA-Analyse

Auf Basis der automatischen SCA-Analyse (s. Abschnitt D6.2) ergeben sich für das Videokonferenz-System Jitsi, nach Abbildung auf unser Gefährdungspotential, folgende Mengen an Findings:



A 3.2 Findings mit ausführlicher Beschreibung in diesem Dokument

In der folgenden Übersicht sind die Bedrohungen und Schwachstellen aufgeführt, welche durch manuelle SAST- bzw. DAST-Ansätze aufgefunden wurden, bzw. bei den automatischen SAST-Analysen von uns mindestens mit dem Gefährdungspotential „hoch“ eingeschätzt wurden.

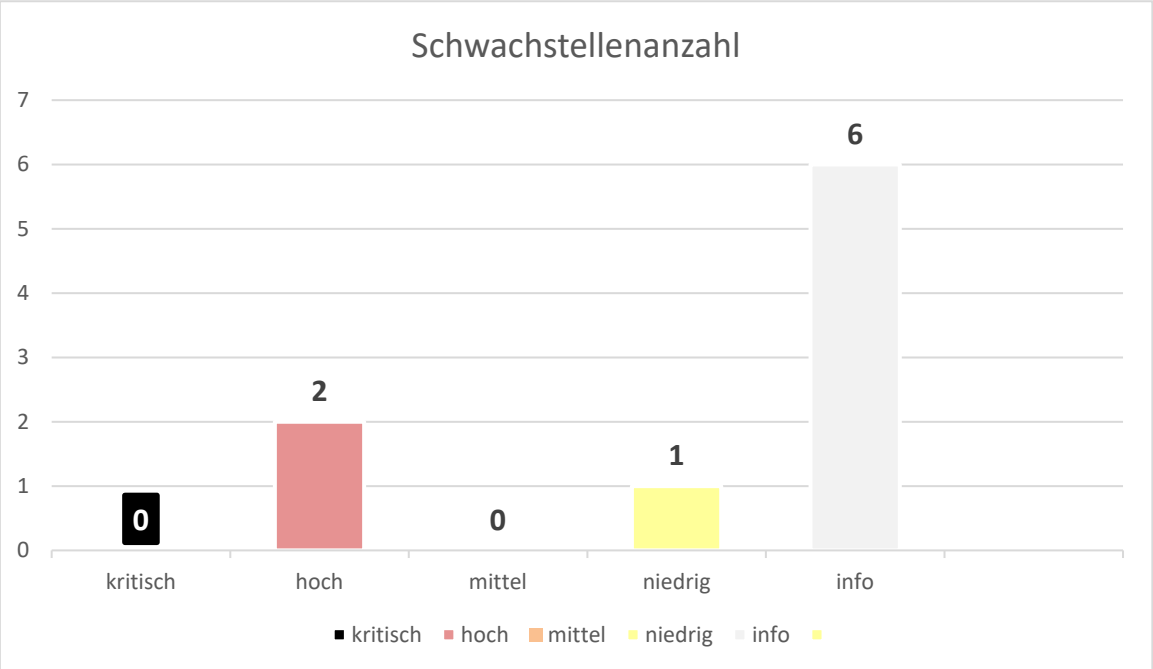
Die folgende Übersicht listet dabei zunächst alle Schwachstellen, welche mit niedrig, mittel, hoch oder kritisch eingeschätzt wurden (Findings im BigBlueButton sind mit dem Prefix „BBB: “ bzw. im Jitsi mit dem Prefix „Jitsi: “ versehen).

Schwachstelle	[niedrig]	[mittel]	[hoch]
Datenvalidierung			
BBB: Stored-Cross-Site-Scripting-Schwachstelle in Bigbluebutton/Greenlight			E1.1.1
BBB: Stored-XSS-Schwachstelle in Bigbluebutton/bigbluebutton-html5			F1.1.1
Authentisierung			
BBB: Brute-Force auf Zugriffcode für Räume in Bigbluebutton/Greenlight möglich	F1.4.1		

In der folgenden Übersicht sind die Beaobachtungen aufgeführt, die das Gefährdungspotenzial [info] besitzen.

[info]	Kapitel
BBB: Verwendung `dangerouslySetInnerHTML` als mögliche XSS-Quelle	E1.1.2
Jitsi: Verdacht auf Cross-Site-Scripting im Shibboleth-Login	E2.1.1
Jitsi: Verwendung `dangerouslySetInnerHTML` als mögliche XSS-Quelle	E2.1.2
BBB: Open Redirect (Selbstangriff) in Bigbluebutton/Greenlight	F1.2.1
BBB: Path Traversal in Bigbluebutton/Greenlight	F1.3.1
BBB: Bekanntgabe von Verifikationstoken-Hashs in Bigbluebutton/Greenlight	F1.5.1

Übersicht



A 4 Assessment Details

A 4.1 Getestete Applikationen

Die Sicherheitsüberprüfung wurde auf folgenden Projekten & Versionen ausgeführt:

A 4.1.1 Jitsi

Name	Package name	Version	Remark	Repo	Branch / Tag / Commit
Jitsi-meet	jitsi-meet	2.0.6865-2	(released 28.01.22)	https://github.com/jitsi/jitsi-meet	T: jitsi-meet_6865
Jitsi_meet_tokens	jitsi-meet-tokens	1.0.5818-1	(released 28.01.22)		
Web config	jitsi-meet-web-config	1.0.5818-1	(released 28.01.22)		
Web	jitsi-meet-web	1.0.5818-1	(released 28.01.22)		
Prosody	jitsi-meet-prosody	1.0.5818-1	(released 28.01.22)		
Turnserver	jitsi-meet-turnserver	1.0.5818-1	(released 28.01.22)		
Jicofo	jicofo	1.0-846-1	(released 28.01.22)	https://github.com/jitsi/jicofo	T: jitsi-meet_6865
Jitsi-videobridge2	jitsi-videobridge2	2.1-617-ga8b39c3f	(released 28.01.22)	https://github.com/jitsi/jitsi-videobridge	T: jitsi-meet_6865
Jibri	jibri	8.0-116-gca0c772-1	(released 17.01.22)	https://github.com/jitsi/jibri	C: ca0c772
Jigasi	jigasi	1.1-216-ga2399b9-1	(released 10.12.21)	https://github.com/jitsi/jigasi	C: a2399b9

Zugrundeliegendes Mengengerüst:

```
    3321 text files.
    3165 unique files.
    1202 files ignored.

github.com/AlDanial/cloc v 1.90  T=5.32 s (573.7 files/s, 80727.2 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
JSON                           140              1              0         100180
JavaScript                    1666          22817          56066          88583
Java                           323           9608          23283          44006
Kotlin                         336           4105          10386          30226
Sass                            94           1182             28           7323
Lua                             39            991           1046           5058
Markdown                        55            725              0           3477
Maven                            8             58            143           2445
SVG                            148              1             58           2430
Objective-C                     26            586            618           1979
Bourne Shell                    56            490            516           1904
XML                             47             90            119           1380
Swift                           17            331            328           1146
Bourne Again Shell              11            272            273           1026
Gradle                           7            131            154            630
HTML                            19             36             29            498
CSV                              1              0              0            428
Python                           2             85             36            248
C/C++ Header                    27            152            530            227
make                             6             53             21            186
DOS Batch                        4             55              4            178
YAML                             8             35             15            152
diff                             6             10             73             61
ProGuard                         2             22             23             54
Perl                             1              7              0             35
D                                1              0              0             10
Windows Resource File            1              1              0              7
INI                              1              0              0              2
-------------------------------------------------------------------------------
SUM:                          3052          41844          93749         293879
-------------------------------------------------------------------------------
```

A 4.1.2 Big Blue Button

Name	Package name	Version	Remark	Repo	Branch / Tag / Commit
BigBlueButton	bigbluebutton	2.4.4	(released 10.02.2022)	https://github.com/bigbluebutton/bigbluebutton	T: v2.4.4
Greenlight	greenlight	2.11.1	(released 14.01.2022)	https://github.com/bigbluebutton/greenlight	T: release-2.11.1
API PHP	bigbluebutton-api-php	2.1.1	(released 24.01.2022)	https://github.com/bigbluebutton/bigbluebutton-api-php	T: 2.1.1
Install	bbb-install	commit cdbf737	(released 05.02.2022)	https://github.com/bigbluebutton/bbb-install	C: cdbf737
Webrtc-sfu	bbb-webrtc-sfu	2.6.9	(released 04.02.2022)	https://github.com/bigbluebutton/bbb-webrtc-sfu	T: v2.6.9
Webhooks	bbb-webhooks	2.5.0	(released 04.02.2022)	https://github.com/bigbluebutton/bbb-webhooks	T: v2.5.0
Playback	bbb-playback	3.1.1	(released 06.10.2021)	https://github.com/bigbluebutton/bbb-playback	T: v3.1.1
Events	bbb-events	1.2.0	(released 19.07.2021)	https://github.com/bigbluebutton/bbb-events	T: 1.2.0
Pads	bbb-pads	1.0.1	(released 21.01.2022)	https://github.com/bigbluebutton/bbb-pads	T: v1.0.1

Zugrundeliegendes Mengengerüst:

```
    3828 text files.
    3646 unique files.
    1659 files ignored.

1 error:
Line count, exceeded timeout:  bbb/bigbluebutton/bbb-api-demo/src/main/webapp/js/jquery.jqgrid.min.js

github.com/AlDanial/cloc v 1.90  T=7.46 s (461.5 files/s, 81374.2 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
JSON                           107            183              0         102448
JavaScript                    1038          16224          17670          99197
XML                            104            646           1638          97072
JSX                            304           5446           1230          43538
CSS                             59           3652           1753          31796
YAML                            95            346           1283          29569
Java                           403           5270           6812          22273
Scala                          437           5051           2464          21947
Ruby                           222           4024           4107          15260
Sass                           156           2255             88          12520
Bourne Shell                   156           1448            718           4646
ERB                             95            273              0           4482
C                                3           1118            590           4434
SVG                             22              6             14           4137
Groovy                          32            759            728           3900
PHP                             68            868           2970           3323
JSP                             24            813            593           3021
Markdown                        36            693              0           2853
Bourne Again Shell              12            467            499           2422
HTML                            30            385            252           2128
Python                           2            110             64            512
Scheme                           6             86              0            381
Grails                           7             20              7            299
Gradle                           7             43             24            262
Maven                            2              0             23            159
diff                             7             45            171            153
DOS Batch                        2             47              4            128
reStructuredText                 1             26              0             88
make                             2              9              3             66
INI                              1              9              0             33
Cucumber                         2              5              0             21
TOML                             1              0              1              2
-------------------------------------------------------------------------------
SUM:                          3443          50327          43706         513070
-------------------------------------------------------------------------------
```

Für die dynamischen Analyseanteile (DAST und IAST) wurden beide Video-Konferenzsysteme auf separierten Testsystemen installiert. Sämtliche (Haupt-) Funktionalitäten wurden durch den Tester dabei manuell getriggert und untersucht (s. auch unsere Methodik-Beschreibung in Abschnitt B).

A 4.2 Testzeitraum

Die Tests wurden im Zeitraum von 01.02.2022 bis 25.04.2022 durchgeführt.

A 4.3 Einschränkende Rahmenbedingungen

A 4.3.1 Quellcodeanalyse

Im Vorfeld der Analyse wurden folgende Festlegungen hinsichtlich der Durchführung getroffen:

- keine spezifische Untersuchung von Vagrant-/Docker-Releases + Konfigurationen
- keine spezifische Untersuchung des Quellcodes von 3rd-Party-Abhängigkeiten
- keine Durchführung von Befragungen/Interviews der Projektbeteiligten (Leiter, Entwickler etc.)
- keine Untersuchung von mobile Code-Anteilen (Android-, iOS-Code etc.)
- keine Untersuchung von Demo- oder Test-Code
- freie Auswahl der freien und kommerziellen Analysewerkzeuge

A 4.3.2 Penetration Test

Im Vorfeld der Analyse wurden folgende Festlegungen hinsichtlich der Durchführung getroffen:

- Module zur dynamischen Analyse werden durch den Auftragnehmer festgelegt
- Installation der Anwendungen durch den Auftragnehmer auf eigener Infrastruktur
- Schwachstellenfokussierung wird durch Auftragnehmer festgelegt (siehe auch die spezifische aufgeführten Bedrohungen in Kapitel A4.5)

Hinweis: Ergebnisse, welche durch diesen Ansatz aufgefunden wurden, können dem Kapitel F entnommen werden.

A 4.4 Toolgestützte Analysen

Beide Anwendungen wurden einer eingehenden Sicherheitsuntersuchung unterzogen. Dafür kam eine Kombination folgender Herangehensweisen zum Einsatz:

- Automatische und manuelle Quellcodeanalyse (SAST + SCA)
- Automatische und manuelle Penetrationstests (DAST)
- DAST-begleitende Analyse durch Einsatz einer IAST-Lösung

Die konkret im jeweiligen Ansatz eingesetzten Werkzeuge werden in den folgenden Abschnitten aufgelistet. Alle Tool-Resultate sind diesem Bericht sowohl in toolspezifischem RAW-Format (falls vorhanden), als auch in Berichtsform (entweder als PDF, Excel, JSON und/oder SARIF) als Anhang (s. auch Abschnitt G) beigefügt.

Alle Ergebnisberichte sind vollständig auditiert und alle Findings wurden bewertet (unsere Bewertungsvorgehen ist in B1.2 beschrieben).

Eine grobgranulare Finding-Übersicht zu den initial vom Werkzeug gelieferten Findings (inkl. ihrer toolspezifischen Kritikalitätseinstufung), sowie zu den durch den Auditor vorgenommenen Bewertungen, sind dem Abschnitt D zu entnehmen.

A 4.4.1 SAST: Eingesetzte Werkzeuge

Die zu analysierenden Module wurden mit folgenden SAST-Werkzeugen einem Scan unterzogen:

- Microfocus Fortify (https://www.microfocus.com/de-de/products/static-code-analysis-sast/overview) - Standard-Regelsatz, Stand März 2022
  - Ergebnisse siehe Kapitel D1
- Checkmarx (https://checkmarx.com/) - Standard-Regelsatz, Stand März 2022
  - Ergebnisse siehe Kapitel D2
- CodeQL (https://codeql.github.com/) - Standard-Regelsatz, Stand März 2022
  - Ergebnisse siehe Kapitel D3
- Semgrep (https://semgrep.dev/) – „p/owasp-top-ten“ Regelsatz, Stand März 2022
  - Ergebnisse siehe Kapitel D4
- TruffleHog (https://github.com/trufflesecurity/trufflehog) - Standard-Regelsatz, Stand April 2022
  - Ergebnisse siehe Kapitel D5

Hinweis: Nicht alle Werkzeuge sind in der Lage, alle Sprachen und Frameworks umfassend zu analysieren. Die Werkzeugauswahl erfolgte durch den Auftragnehmer, jeweils passend zu den verwendeten Programmiersprachen und Frameworks.

A 4.4.2 SCA: Eingesetzte Werkzeuge

Die Abhängigkeiten der zu überprüfenden Module wurden mit dem folgenden SCA-Werkzeug einer Known-Vulnerability-Prüfung unterzogen:

- OWASP DependencyCheck (https://github.com/jeremylong/DependencyCheck) – Known-Vulnerability-Datenbankstände vom 22.04.2022

A 4.4.3 DAST: Eingesetzte Werkzeuge

Die Burp Suite Professional (https://portswigger.net/burp/pro) kam für die dynamische Analyse der zwischen Server und Client ausgetauschten Daten zum Einsatz. Sämtliche (Haupt-) Funktionalitäten wurden hierbei durch den Tester manuell ausgelöst und untersucht. Hierbei kam der „Burp Web Vulnerability Scanner“ (aktiv und passiv) umfassend zum Einsatz.

Alle darüber hinaus vom Tester als „aus Sicherheitssicht interessant“ eingestuften Requests wurden manuell (mit automatischer Werzkeug-Unterstützung) tiefgreifender analysiert. Dabei kam (neben den eingebauten Hilfstools, wie Repeater oder Intruder) auch die Erweiterung „Autorize“ (https://portswigger.net/bappstore/f9bbac8c4acf4aefa4d7dc92a991af2f) zum Einsatz.

Im Datei-Anhang sind die vollständigen Arbeitsprotokolle hinterlegt:

- burp/BBB-burp-protocol.zip
    - Burp-Protokoll für Analysen gegen die BigBlueButton-Umgebung
- burp/Jitsi-burp-protocol.zip
    - Burp-Protokoll für Analysen gegen die Jitsi-Umgebung

Hinweis: Für die in A4.4.4 gelisteten Module erfolgte dabei im Hintergrund eine begleitende IAST-Analyse. Alle durch Burp bzw. durch Browser-Aktionen getriggerte Aufrufe (manuell und automatisch), wurden dabei automatisch von Contrast analysiert.

Hinweis: Gewonnene Erkenntnisse, welche nach unseren Auswertungen ggf. zu validen Findings/Schwachstellen führten, werden für dieses Werkzeug nicht explizit im Anhang als RAW-Format oder Report aufgeführt, sondern führen zu Finding-Beschreibungen in Kapitel F.

A 4.4.4 IAST: Eingesetzte Werkzeuge

Während der (automatisch und manuell durchgeführten) dynamischen Analysen kam bei folgenden Modulen das Werkzeug Contrast Security IAST (https://www.contrastsecurity.com/) zum Einsatz:

- Bigbluebutton Greenlight: Ruby, Contrast agent v.5.3.0
- Bigbluebutton bbb-webhooks: NodeJS, Contrast agent v.4.10.6
- Bigbluebutton bbb-webrtf-sfu: NodeJS, Contrast agent v.4.11.0
- Jitsi Jicofo: Java, Contrast agent v.3.11.0

Hinweis: Bei allen anderen Modulen waren die technischen Voraussetzungen für eine Analyse mit Contrast IAST nicht gegeben.

Im Datei-Anhang sind die RAW Tool-Auswertungen/-Informationen hinterlegt:

- contrast/bbb-findings-and-protocol.xml
    - Contrast-Informationen zu den o.g. BigBlueButton-Modulen
- contrast/jitsi-findings-and-protocol.xml
    - Contrast-Informationen zu den o.g. Jitsi-Modulen

Hinweis: Gewonnene Erkenntnisse, welche nach unseren Auswertungen ggf. zu validen Findings/Schwachstellen führten, werden für dieses Werkzeug nicht explizit aufgelistet, sondern führten zu Finding-Beschreibungen in Kapitel F.

A 4.5 Manuelle Analysen

Im Rahmen der Untersuchungen werden ebenfalls umfangreiche manuelle Code-Reviews durchgeführt, welche aufgrund der LoC (insgesamt ca. 800.000) nicht vollumfassend möglich waren. Es wurden daher zielgerichtete Untersuchungen an neuralgischen Stellen durchgeführt. Folgende, gemeinsam mit dem Auftraggeber initial festgelegten Priorisierungen und Fokussierungen lagen dabei zugrunde:

- Review der CVEs (beginnend 2020) zu den analysierten Konferenzsystemen und Bewertung der Art und Weise des Umgangs mit den gemeldeten Schwachstellen, sowie Analyse der vom Projekt gelieferten Korrektur (siehe eigenständiges Kapitel C)
- Spezifische Bedrohungen:
    - Browserbasierten Bedrohungen/Angriffen (z.B. Session-Fixation, Cross-Site-Scripting)
    - Angriffe auf Vertraulichkeit und Integrität (z.B. Belauschen von Audio-, Video-, Chat- oder Desktopsharing-Daten)
    - Bedrohungen/Angriffe auf den Server (z.B. Command- oder SQL-Injection)
    - Angriffe auf die Nutzerauthentisierung (falls vorhanden)
    - Schutz der Privatsphäre (z.B. Abfluss von vertraulichen Meta-Daten)

Hinweis: Bei der Bewertung der mit den automatischen Werkzeugen aufgezeigten Findings (s. Kapitel D) schaut sich der Auditor auch immer den „umgebenden Code“ mit an und erkennt so ggf. anderweitige Schwachstellen oder sogenannte „Code-Smells“. Er lernt dabei, wie die einzelnen Entwickler programmieren, welche Vorlieben sie haben und welche Fehler man von ihnen „erwarten“ könnte. Die Bewertung von diesen Findings ist somit auch immer eine Form von „geführter manueller Analyse“.

B. Methodik und Bewertung

B 1.1 Bewertungsschema

Die durchgeführten Tests wurden jeweils mit einem Gefahrenpotenzial bewertet. Das Gefahrenpotenzial ist ein abstraktes Maß für die Klassifizierung einer Schwachstelle (Vulnerability) und der durch sie entstehenden Bedrohung (Threat).

Das Gefahrenpotenzial wird zu jeder getesteten Schwachstelle in der rechten Spalte genannt. Wir unterscheiden:

[OK]	die genannte Schwachstelle liegt nicht vor
[kritisch]	Es besteht sofortiger Handlungsbedarf; der Applikation, der Datenbank oder dem System kann schwerwiegender Schaden zugefügt werden; ggf. ist das System abzuschalten
[hoch]	Es besteht dringender Handlungsbedarf, die Webanwendung/das System und/oder die Daten sind bestandsgefährdet
[mittel]	Es besteht Handlungsbedarf, die Webanwendung/das System muss z. B. durch einspielen von Sicherheitsupdates oder durch zusätzliche Sicherungsmaßnahmen (z. B. WAF) abgesichert werden
[niedrig]	Der Handlungsbedarf liegt im Ermessen des Auftraggebers
[info]	hierbei handelt es sich um Informationen, nicht um eine Schwachstelle

Das Gefahrenpotenzial ist kein Maß für das Risiko, das eine solche Schwachstelle darstellt. Das Risiko (= Schadenshöhe * Eintrittswahrscheinlichkeit), welches dieses Gefahrenpotenzial darstellt, wird hier nicht bewertet, da zu dessen Beurteilung weitere Informationen (z. B. Schadenspotenzial) vom Auftraggeber nötig sind.

B 1.2 Aufbau der Testergebnisse der toolbasierten Findings

Alle von den Werkzeugen gemeldetet Schwachstellen wurden bewertet und Ergebnisberichte (falls möglich bzw. notwendig) liegen im Anhang (s. jeweiliges Werkzeug in Abschnitt D bzw. grobe Übersicht in Abschnitt G) bei.

Die tooleigenen Kritikalititätseinstufungen wurden von uns folgendermaßen interpretiert:

- **Kritisch** (in den Tools: „Critical“)
- **Hoch** (in den Tools: „High“ oder „Error“)
- **Mittel** (in den Tools: „Medium“, „Moderate“ oder „Warning“)
- **Niedrig** (in den Tools: „Low“ oder „Recommendation“)

Die eigentlichen Bewertungen erfolgten entweder in der vom Werkzeug bereitgestellten Oberfläche oder in von uns aus den RAW-Ergebnissen erstellten Excel-Dokumenten. Dabei musste bei der Bewertung teilweise auf tool-eigenes „Wording“ zurückgegriffen werden. Die Bewertungen in den Reports sind folgendermaßen zu interpretieren:

- **Bestätigt** (in den Bewertungen: „Exploitable“ oder „Confirmed“)
    - Finding wurde bestätigt (mit „kritisch“ oder „hoch“ bewertete Findings wurden hinsichtlich ihrer Ausnutzbarkeit untersucht und dann in Abschnitt E näher beschrieben)
- **Verdächtig** (in den Bewertungen: „Suspicious“ oder „Proposed not Exploitable“)
    - Finding konnte nicht 100%ig bestätigt werden, könnte aber ein Sicherheitsproblem darstellen. Hier muss die Bewertung von einem Projekt-Insider mit Sicherheitshintergrund abgeschlossen werden.
- **Schlechte Praxis** (in den Bewertungen: „Bad Practice“ oder „Bug“)
    - Finding wurde als Programmierfehler oder schlechte Code-Qualität eingestuft, welcher z.B. sicherheitsrelevante Konsequenzen oder logische Fehlinterpretationen zur Folge haben könnte. Die Wahrscheinlichkeit dafür wurde aber als sehr gering eingestuft.
- **Kein Problem** (in den Bewertungen: „Not an issue“ oder „False Positive“)
    - Finding wurde als irrelevant oder False Positive eingestuft.

Auf Basis dieser beiden Einstufungen ergibt sich die folgende Interpretation zum in A1.1.1 definierten Gefahrenpotential (die Statistik-Übersicht in A3.1 basiert auf dieser Zuordnung):

[kritisch]

- Tooleinstufung: kritisch X Auditoreinstufung bestätigt

[hoch]

- Tooleinstufung: hoch X Auditoreinstufung bestätigt

[mittel]

- Tooleinstufung: kritisch X Auditoreinstufung verdächtig
- Tooleinstufung: mittel X Auditoreinstufung bestätigt

[niedrig]

- Tooleinstufung: kritisch X Auditoreinstufung schlechte Praxis
- Tooleinstufung: hoch X Auditoreinstufung verdächtig
- Tooleinstufung: niedrig X Auditoreinstufung bestätigt

[info]

- Tooleinstufung: hoch X Auditoreinstufung schlechte Praxis
- Tooleinstufung: mittel X Auditoreinstufung verdächtig
- Tooleinstufung: mittel X Auditoreinstufung schlechte Praxis
- Tooleinstufung: niedrig X Auditoreinstufung verdächtig
- Tooleinstufung: niedrig X Auditoreinstufung schlechte Praxis

C. CVE-Reviews

Im Rahmen dieser Analyse wurden für BigBlueButton (s. C1) und Jitsi (s. C2) alle gemeldeten CVEs (seit 2020) hinsichtlich Ihrer Korrektur überprüft und kommentiert.

CVE	Inhalt	Kommentar
BBB		
CVE-2022-27238	XSS im Usernamen (Chat)	(noch) nicht mitigiert (Noch nicht veröffentlicht)
CVE-2022-26497	XSS im Usernamen (Share Room Access)	Korrektur i.O. (Noch nicht veröffentlicht)
CVE-2021-4143	XSS im Usernamen (geteilte Notizen) über Fork	Korrektur i.O.
CVE-2020-29043	Registrierungstoken wurde geleaked	Auslieferung vom Hash ermöglicht weiterhin Offline Brute-Force Angriffe
CVE-2020-29042	Brute-Force auf Zugangscode möglich	Nicht mitigiert
CVE-2020-28954	Angriffe auf Interpreter durch Steuerzeichen möglich	Korrektur i.O.
CVE-2020-28953	Mehrfach-Voting möglich	Korrektur i.O.
CVE-2020-27642	XSS in Merge-Account Funktion	Korrektur i.O.
CVE-2020-27613	Verwendung Default-Passwort für Freeswitch	Korrektur i.O.
CVE-2020-27612	Nutzernamen in Raum-URL geleaked	Teile vom Namen immernoch in URL
CVE-2020-27611	Verwendung Freeswitch STUN/TURN-Server	Switch auf Google ist zumindest aus Privacy-Sicht fraglich.
CVE-2020-27610	Netzwerkservices während der Installation extern erreichbar	Command-Option verhindert, jedoch kein „Secure Defaults“
CVE-2020-27609	Session-Aufzeichnung immer aktiv	Wird heute immernoch per default aufgezeichnet – Privacy!
CVE-2020-27608	Fehlender Mime-Type bei geteilten Daten (XSS)	Korrektur i.O.
CVE-2020-27607	Software-Mic sendet Daten auch bei mute	Korrektur i.O.
CVE-2020-27606	Fehlendes secure-Flag am Session-Cookie	Korrektur i.O.
CVE-2020-27605	Injection-Angriffe auf Ghostscript-/LibreOffice-Verarbeitung möglich	Korrektur i.O.
CVE-2020-27604	Injection-Angriffe auf Ghostscript-/LibreOffice-Verarbeitung möglich	Korrektur i.O.
CVE-2020-27603	Einbinden von externen Dateien bei Ghostscript-/LibreOffice-Verarbeitung möglich	Korrektur i.O.
CVE-2020-26163	URLs in Registrierungsmails können gespofft werden	Korrektur i.O.
CVE-2020-25820	SSRF zum Auslesen von Serverdaten missbrauchbar	Korrektur i.O.
CVE-2020-12443	Serverseitiges Path-Traversal möglich	Korrektur i.O.
CVE-2020-12113	XSS durch dangerouslySetInnerHTML	Korrektur i.O.
CVE-2020-12112	Diebstahl lokaler Dateien durch serverseitige File-Inclusion	Korrektur i.O.

Jitsi		
CVE-2021-26812	XSS in Erweiterung	Out-of-Scope (Erweiterung/Mod)
CVE-2020-25019	Ausführung eines beliebigen Executables auf dem Server möglich	Out-of-Scope (Erweiterung/Mod)
CVE-2020-11878	Default-/Trivial-Passwörter kommen zum Einsatz	Zugehörige Commit-Umsetzung schlecht, jedoch heute solide umgesetzt

Folgende erwähnenswerte, übergreifende Beobachtung wurden bei der Korrektur-Bewertung beider Konferenzsysteme gemacht:

- es wurden für die überprüften Korrekturen keine Unit- oder Integrationstests angelegt bzw. angepasst

C 1 BigBlueButton

Stand 17.03.2022 (Übersicht:
https://www.cvedetails.com/vendor/23257/Bigbluebutton.html).

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
2020	21						3	1			3			1	
2022	1						1								
Total	22						4	1			3			1	
% Of All		0.0	0.0	0.0	0.0	0.0	18.2	4.5	0.0	0.0	13.6	0.0	0.0	4.5	

Hinweis: Bezugnehmend auf
https://github.com/bigbluebutton/bigbluebutton/labels/target%3A%20security
(bigbluebutton-issues mit dem Tag „target: security“) gibt es noch weitere offen Security-Issues, welche bis ins Jahr 2015 zurückreichen, bisher aber nicht final bearbeitet wurden.

Folgende erwähnenswerte, übergreifende Beobachtungen wurden bei der Korrektur-Bewertung gemacht:

- in den Commit- bzw. Release-Kommentaren ist manchmal nur schwer oder überhaupt nicht erkennbar, dass bestimmte CVEs korrigiert wurden
- relevante Commits sind aus dem CVE-Eintrag manchmal nicht verlinkt
- einige Fixes wirken etwas unorganisiert (z.B. Änderungen werden übernommen und dann etwas später mit einem anderen Lösungsansatz überschrieben)
- bestimmte Entscheidungen können aus Sicherheitssicht nicht 100% nachvollzogen werden (z.B. Verwendung STUN-Server von Google, Recording-Verhalten)
- das generelle Herangehen erscheint manchmal fragwürdig (z.B. die Einbindung von eigenen Github-Repositories als Fork von offziellen Bibliotheken ins BBB-Produktivsystem)

Insgesamt wurde der Gesamteindruck aber besser, je jünger die jeweils behandelte CVE war.

C 1.1 CVE-2022-27238 (noch nicht veröffentlicht)

C 1.1.1 Steckbrief

Fix-Bewertung	Stand 24.04.2022: nicht mitigiert
CWE	79: Failure to Preserve Web Page Structure ('Cross-site Scripting')
CVE-Kurzbeschreibung	A threat actor could inject a JavaScript payload into their username. The payload gets executed in the browser of the victim in the "Private chat" dialog every time the attacker sends a private message to the victim.
Veröffentlichung	21.03.2022

CVE-Details	https://www.cvedetails.com/cve/2022-27238/
CVSS-Score	TBD
Finder-Beschreibung	TBD
Zusammenfassung	XSS im Nutzernamen kommt im privaten Chat zur Ausführung
Betroffene Versionen	< 2.5

C 1.1.2 Korrektur-Informationen

Noch nicht mitigiert.

C 1.1.3 Bewertung

Noch nicht mitigiert.

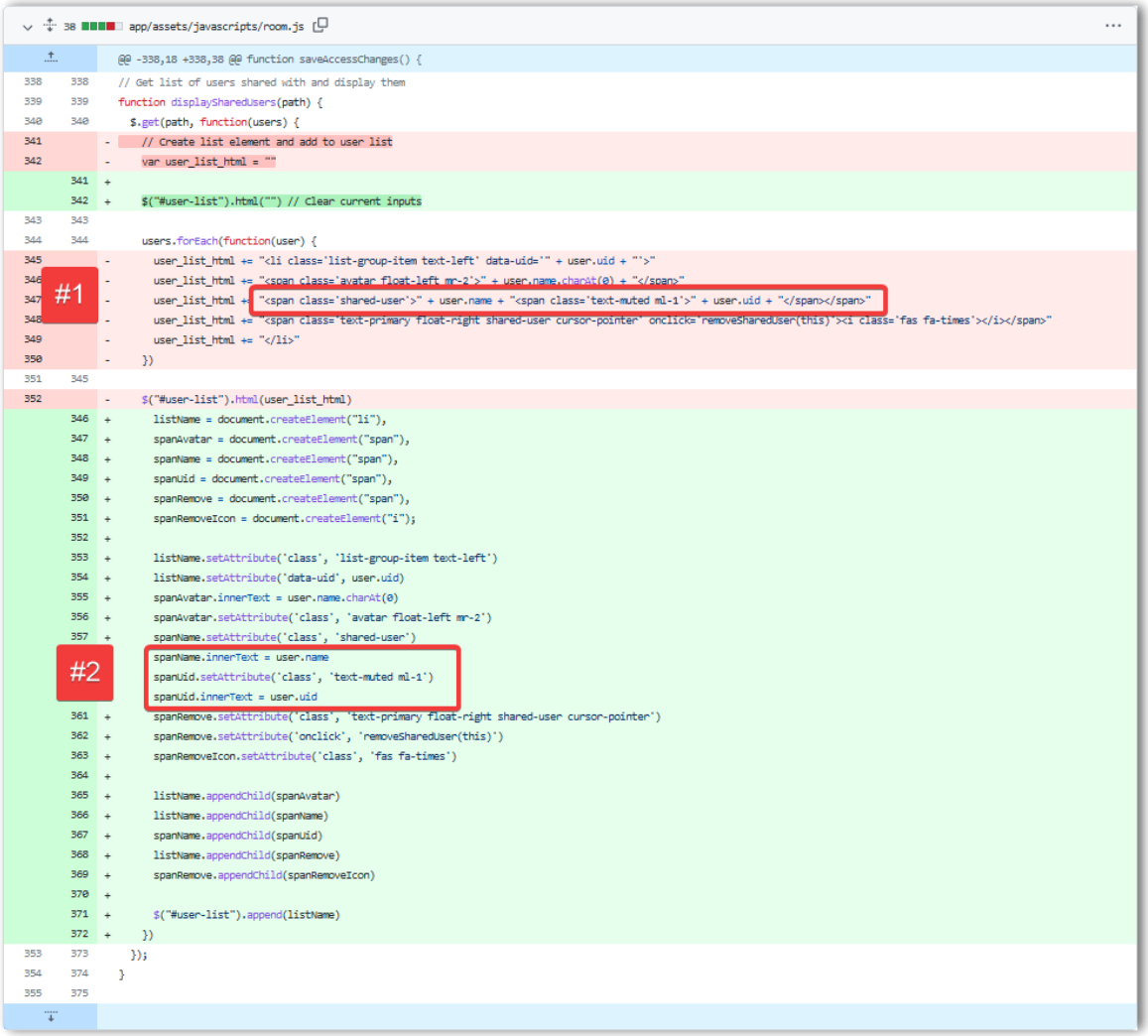
C 1.2 CVE-2022-26497 (noch nicht veröffentlicht)

C 1.2.1 Steckbrief

Fix-Bewertung	i.O.
CWE	79: Failure to Preserve Web Page Structure ('Cross-site Scripting')
CVE-Kurzbeschreibung	A threat actor could inject JavaScript payload in his/her username. The payload gets executed in the browser of the victim in the "Share Room Access" dialog if the victim has shared access to the room with the attacker previously.
Veröffentlichung	04.03.2022
CVE-Details	https://www.cvedetails.com/cve/2022-26497/
CVSS-Score	TBD
Finder-Beschreibung	TBD
Zusammenfassung	XSS im Nutzernamen kommt bei der Raum-Zugriffsverwaltung zur Ausführung
Betroffene Versionen	< 2.11.1

C 1.2.2 Korrektur-Informationen

Im zugehörigen Commit
https://github.com/bigbluebutton/greenlight/commit/547b841b0e178e06bc2d104cca5cc81e2a542e1e wird das alte Text-zu-HTML-Vorgehen (es wird vom Browser ein konkatenierter String geparst) an der betroffenen Stelle (#1) durch direkte DOM-Manipulation mit sicheren API-Aufrufen (#2) ersetzt:



C 1.2.3 Bewertung

i.O.

C 1.3 CVE-2021-4143

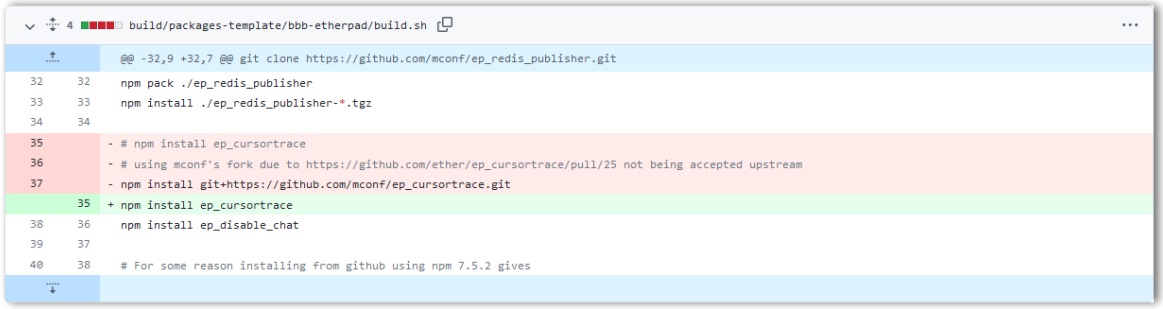
C 1.3.1 Steckbrief

Fix-Bewertung	i.O.
CWE	79: Failure to Preserve Web Page Structure ('Cross-site Scripting')
CVE-Kurzbeschreibung	Cross-site Scripting (XSS) - Generic in GitHub repository bigbluebutton/bigbluebutton prior to 2.4.0.
Veröffentlichung	19.01.2022
CVE-Details	https://www.cvedetails.com/cve/CVE-2021-4143/
CVSS-Score	4.3

Finder-Beschreibung	https://huntr.dev/bounties/e67603e6-8497-4ab6-b93a-02c26407d443/
Zusammenfassung	XSS im Nutzernamen kommt in den „geteilten Notizen“ bei anderen Nutzern zur Ausführung
Betroffene Versionen	< 2.4.0

C 1.3.2 Korrektur-Informationen

Der referenzierte Fix (https://github.com/bigbluebutton/bigbluebutton/commit/62040bdcb3c2f993ba72ab89f4db2015e18d1706) korrigiert die direkte Einbindung eines Forks (https://github.com/mconf/ep_cursortrace) einer Bibliothek, welche die Cursor-Position in Etherpad anzeigt (https://github.com/ether/ep_cursortrace) zurück auf das offizielle Release. Dafür wurde folgende Korrektur in der Datei `build.sh` vorgenommen:

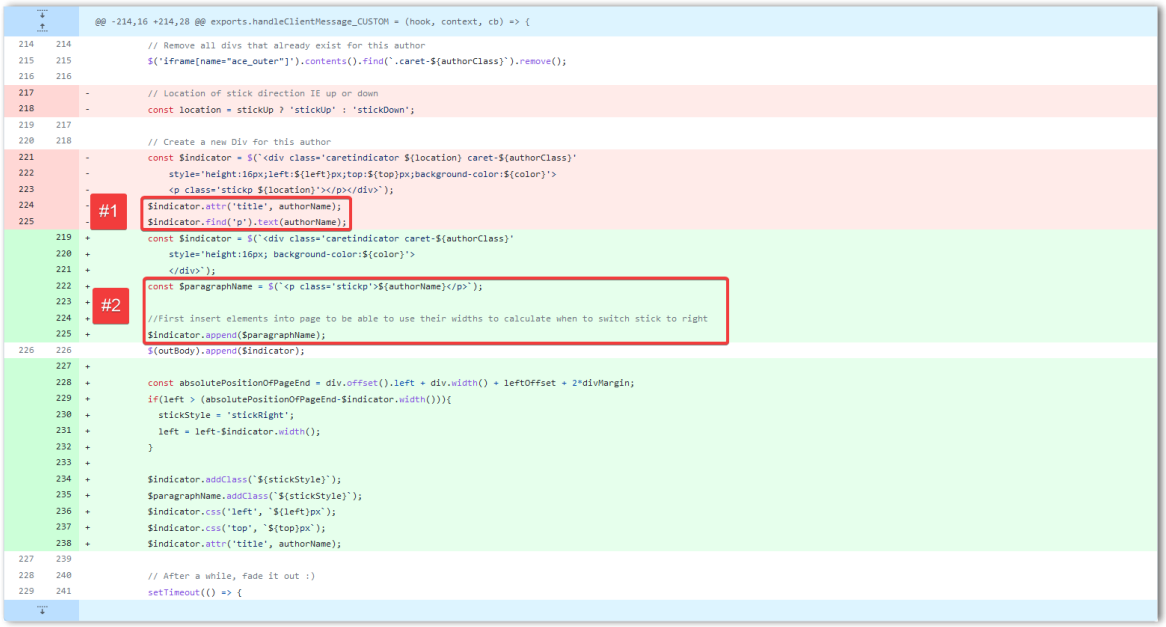


C 1.3.3 Bewertung

i.O.

Grundsätzlich kann man sich hier fragen, wie es sein kann, dass ein veränderter, inoffzieller Fork im produktiven BigBlueButton-Release zum Einsatz kam. Der im Screenshot oben ersichtliche PULL-Request, welcher offensichtlich der Auslöser dieser „Operation“ war, da er vom offiziellen Projekt wohl nicht akzeptiert wurde, korrigiert die Position des am Cursor angezeigten Nutzernamens auf „links vom Cursor“, sobald der Cursor zu dicht am rechten Rand angelangt ist (im Original verschwindet der Name).

Schaut man sich die Änderungen im PULL-Request (in der Datei `static/js/main.js`) an, erkennt man den zugrundeliegenden Fehler recht gut (https://github.com/ether/ep_cursortrace/pull/25/commits/8e13a10b1cac85efe104ebe96adeff9a5f8a9406):



Im Original wird an #1 der Nutzername über die JQuery-Funktionen `.attr()` bzw. `.text()` in das Indicator-`<div>` bzw. das enthaltene `<p>`-Element eingefügt. Beide Funktionen verarbeiten den Payload als Text und lassen diese nicht vom Parser verarbeiten. Dadurch wird an diesen Stellen XSS verhindert.

Im inoffziellen Fork wird an #2 mit JavaScript-Templates gearbeitet, wodurch der Payload erstmal direkt in den Template-String übernommen wird. Da das Template selbst Markup enthält, wird das Ergebnis dann im Anschluss über die parsende JQuery-Funktion `append()` in das umgebende `<div>`–Element eingebunden. Dadurch wurde das XSS an dieser Stelle ermöglicht.

C 1.4 CVE-2020-29043

C 1.4.1 Steckbrief

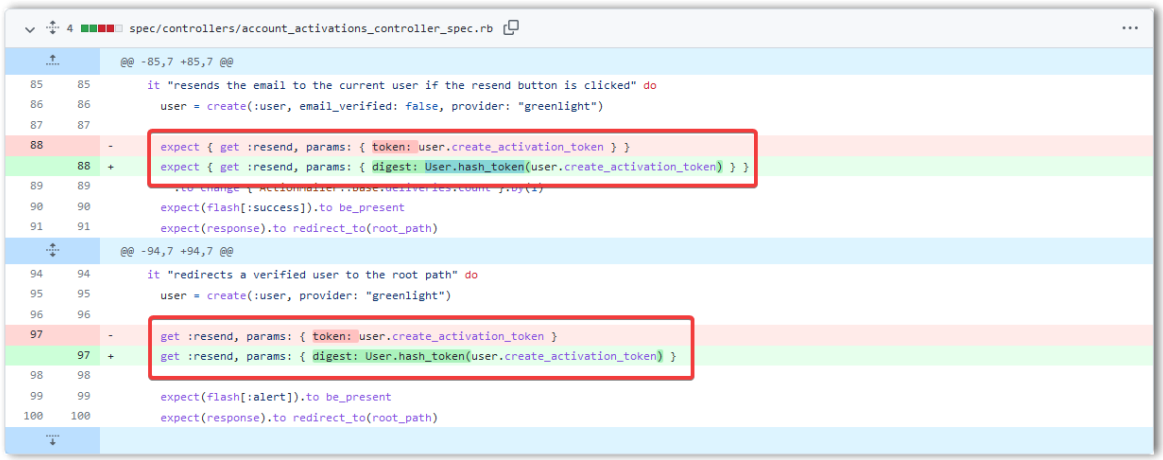
Fix-Bewertung	Auslieferung vom Hash ermöglicht weiterhin Offline-Bruteforce (s. F1.5.1)
CWE	CWE-862: Missing Authorization
CVE-Kurzbeschreibung	An issue was discovered in BigBlueButton through 2.2.29. When an attacker is able to view an account_activations/edit?token= URI, the attacker can create an approved user account associated with an email address that has an arbitrary domain name.
Veröffentlichung	26.11.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-29043/
CVSS-Score	5.0
Finder-Beschreibung	https://cxsecurity.com/issue/WLB-2020110211

Zusammenfassung	Es kann eine beliebige E-Mail registriert und anschließend approved werden, da der E-Mail-Authorization-Token Teil des anschließenden Aufrufes ist und so vom Angreifer einfach ausgelesen werden kann. Dieser Umstand kann z.B. für Social-Engineering Angriffe verwendet werden.
Betroffene Versionen	<= 2.2.29

C 1.4.2 Korrektur-Informationen

Der CVE-Eintrag enthält keine genaue Commit-Information, sondern lediglich einen Verweis auf die Releases-Seite (https://github.com/bigbluebutton/bigbluebutton/releases).

Unsere Recherchen deuten darauf hin, dass das Problem mit dem Commit https://github.com/bigbluebutton/greenlight/commit/95b86b167e75b543036bc16cf854a9a9795e2b75 mitigiert wurde:



Der Token wird als 16 Zeichen langer Zufallswert über `SecureRandom.urlssafe_base64()` (https://apidock.com/ruby/SecureRandom/urlsafe_base64/class) generiert (Zeichenvorrat: 64 bestehend aus A-Z, a-z, 0-9, “-” and “_“). Dieser wird per Mail an die registrierende Entität/E-Mail gesendet.

In der `User`-Klasse wird über die `hash_token()`-Funktion dann ein SHA2-256 erzeugt und serverseitig abgespeichert. Der Hash-Wert wird dem registrierenden Nutzer in einem GET-Response übermittelt, damit dieser diesen später über den Browser wieder referenzieren kann.

C 1.4.3 Bewertung

Ein 16 Zeichen langer `SecureRandom`-Wert weist genügend Entropie auf, um einem Brute-Force-Angriff recht lange stand zu halten. Dennoch ist das „Leaken“ des Hashes nicht die idealste Herangehenweise, um die Registrierung zu realisieren, zumal hier Offline mit BruteForce-Techniken oder (partiellen) Rainbow-Tables gearbeitet werden kann.

C 1.5 CVE-2020-29042

C 1.5.1 Steckbrief

Fix-Bewertung	Nicht mitigiert (s. F1.4.1)
CWE	307: Improper Restriction of Excessive Authentication Attempts
CVE-Kurzbeschreibung	An issue was discovered in BigBlueButton through 2.2.29. A brute-force attack may occur because an unlimited number of codes can be entered for a meeting that is protected by an access code.
Veröffentlichung	26.11.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-29042/
CVSS-Score	4.3
Finder-Beschreibung	https://cxsecurity.com/issue/WLB-2020110210
Zusammenfassung	Brute-Force Angriff auf Zugangscode möglich (Meeting-Link muss bekannt sein)
Betroffene Versionen	<= 2.2.29

C 1.5.2 Korrektur-Informationen

Der CVE-Eintrag enthält leider keine genau Commit-Information, sondern lediglich einen Verweis auf die Releases-Seite (https://github.com/bigbluebutton/bigbluebutton/releases).

Entsprechend der Zeit- und Versionsangaben in der CVE ist es am wahrscheinlichsten, dass das Problem mit dem 2.2.30 Release (https://github.com/bigbluebutton/bigbluebutton/releases/tag/v2.2.30) gefixt worden ist.

C 1.5.3 Bewertung

Dass das Projekt bei keinem seiner Releases im zeitlichen passenden Bereich (geprüft: 2.2.29 – 2.3.35) auf einen passenden Security-Impact und –Fix hinweist, wirft Fragen auf. Es gibt in den o.g. Releases Hinweise auf Security-Fixes, aber keines scheint diese CVE zu referenzieren, oder vom Text her zu passen. Vermutlich ist die Korrektur im Rahmen einer anderen Anpassung erfolgt.

Unsere Rechereche in den Security-relevanten Closed-Issues (https://github.com/bigbluebutton/bigbluebutton/issues?q=label%3A%22target%3A+security%22+is%3Aclosed) lieferte leider auch keine neuen Erkenntnisse hinsichtlich des entsprechenden Commits.

Hinweis: Dieses Problem besteht in der gesteten Version immernoch und wurde offenbar nicht mitigiert.

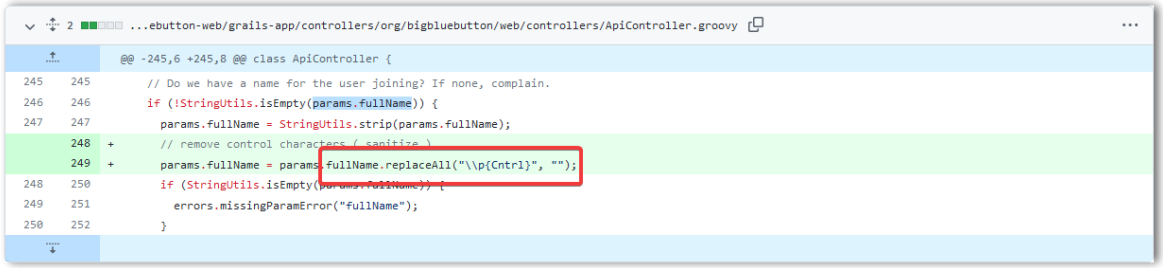
C 1.6 CVE-2020-28954

C 1.6.1 Steckbrief

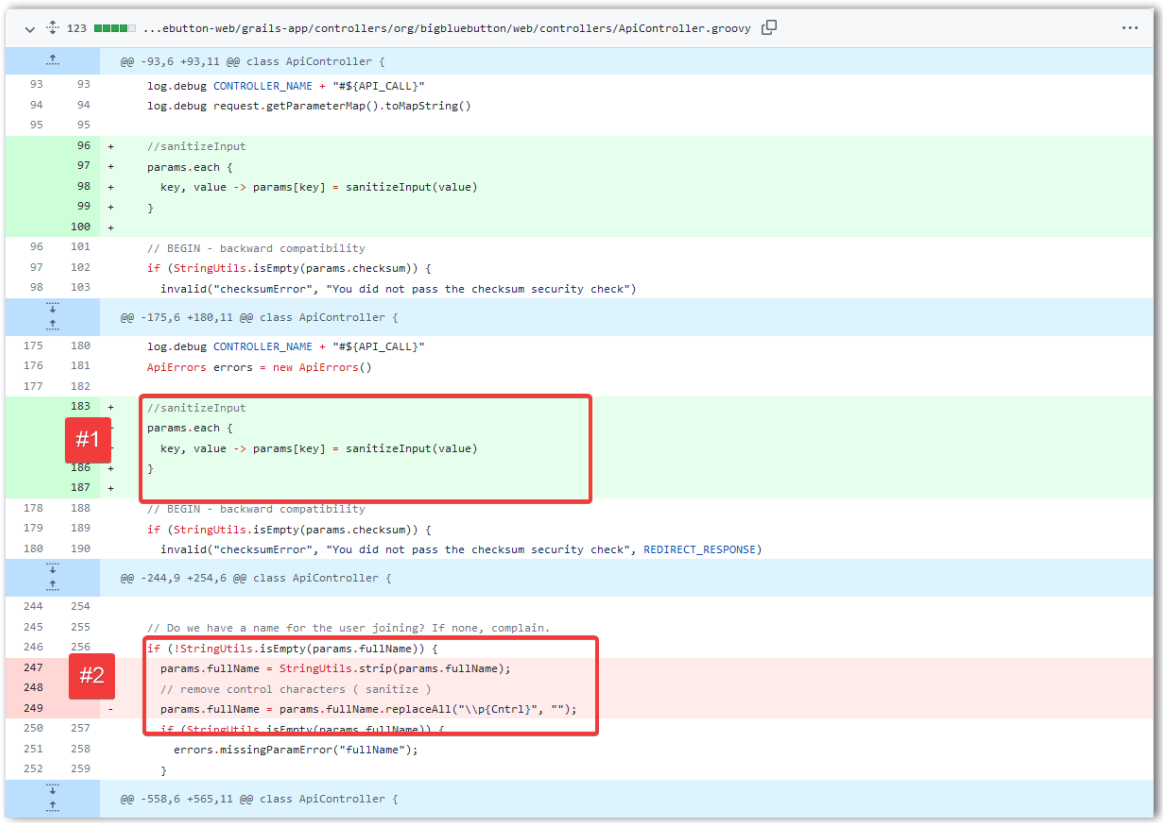
Fix-Bewertung	i.O.
CWE	116 : Improper Encoding or Escaping of Output
CVE-Kurzbeschreibung	web/controllers/ApiController.groovy in BigBlueButton before 2.2.29 lacks certain parameter sanitization, as demonstrated by accepting control characters in a user name.
Veröffentlichung	29.11.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-28954/
CVSS-Score	5.0
Finder-Beschreibung	https://github.com/bigbluebutton/bigbluebutton/issues/10818
Zusammenfassung	Join-API erlaubt spezielle Steuerzeichen, welche z.B. für nachgelagerte Interpreter (z.B. XML-Parser) ein Problem werden können und unnötigerweise akzeptiert werden.
Betroffene Versionen	< 2.2.29

C 1.6.2 Korrektur-Informationen

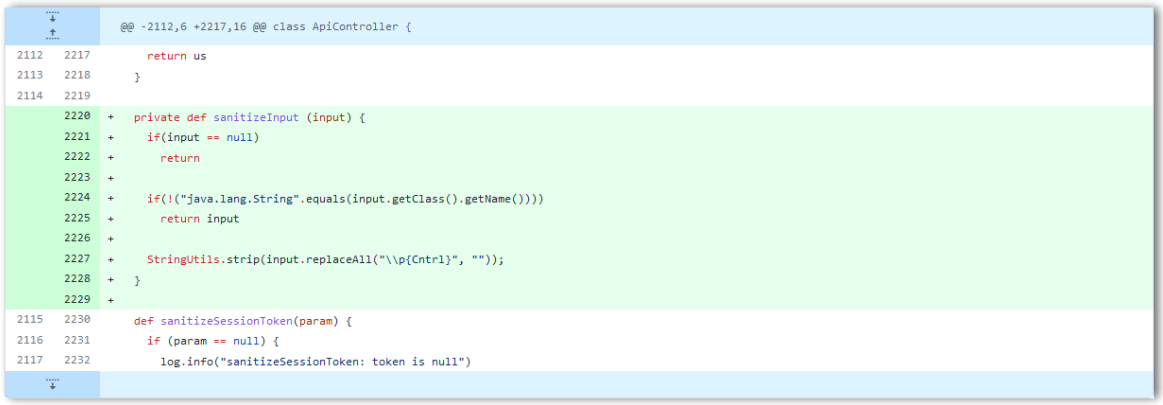
Der CVE-Eintrag verweist auf 2 Commits, wovon der erste Commit (https://github.com/bigbluebutton/bigbluebutton/commit/5c911ddeec4493f40f42e2f137800ed4692004a4) eine einzelne Stelle mit einer Steuerzeichenersetzung versieht:



und der zweite Commit (https://github.com/bigbluebutton/bigbluebutton/commit/e59bcd0c33a6a3203c011faa8823ba2cac1e4f37) anscheinend etwas systematischer an 20 API-Endpunkten mit einem generalisierten Funktionsaufruf (#1) versieht und den ersten Commit wieder zurücksetzt (#2):



Die `santizeInput()`-Funktion ist dabei wie folgt definiert:



Es wird bei der Behandlung mit dem Pattern `\\p{Cntrl}` gearbeitet, welches sich auf „ASCII or Latin-1 control character: 0x00–0x1F and 0x7F–0x9F“: https://www.regular-expressions.info/unicode.html#category.

Hinweis: Weitere Steuerzeichen und Code-Points, wie „ungenutze“ oder Surrogates werden hier explizit aus der Ersetzung ausgeschlossen. Dies ist auch in der heutigen Implementierung noch der Fall, jedoch wurde die Behandlung über den `ValidationService` in die Klasse `ParamsUtil` ausgelagert:

```
12
13  public class ParamsUtil {
14    private static Logger log = LoggerFactory.getLogger(ParamsUtil.class);
15
16    private static final Pattern VALID_ID_PATTERN = Pattern.compile("[a-zA-Z][a-zA-Z0-9- ]*$");
17
18    public static final String INVALID_CHARS = ",";
19
20    public static String stripControlChars(String text) {
21      return text.replaceAll("\\p{Cc}", "");
22    }
23
24    public static String escapeHTMLTags(String value) {
```

C 1.6.3 Bewertung

Grundsätzlich deckt das Pattern `\\p{Cc}` einen sehr großen Bereich der potentiell gefährlichen Steuerzeichen ab. Eine Verwendung eines Interpreters, der andere Steuerzeichen, wie „unsichtbare Formatierungsindikatoren“ oder „Surrogates“ als Data2Code Begrenzer verwendet, ist höchst unwahrscheinlich. Daher i.O.

Hinweis: Die alte `sanitizeInput()`-Funktion existiert heute noch, ist jedoch mit dem Kommentar „`// Can be removed. Input sanitization is performed in the ValidationService.`“ Markiert. Verwendet wird sie vom API-Endpunkt `uploadDocuments()`:

```
1214  def uploadDocuments(conf) { //
1215    log.debug("ApiController#uploadDocuments(${conf.getInternalId()})");
1216
1217    //sanitizeInput
1218    params.each {
1219      key, value -> params[key] = sanitizeInput(value)
1220    }
1221
1222    String requestBody = request.inputStream == null ? null : request.inputStream.text;
1223    requestBody = StringUtils.isEmpty(requestBody) ? null : requestBody;
```

Hier wurde offenbar noch nicht vollständig auf den `ValidationService` umgestellt.

C 1.7 CVE-2020-28953

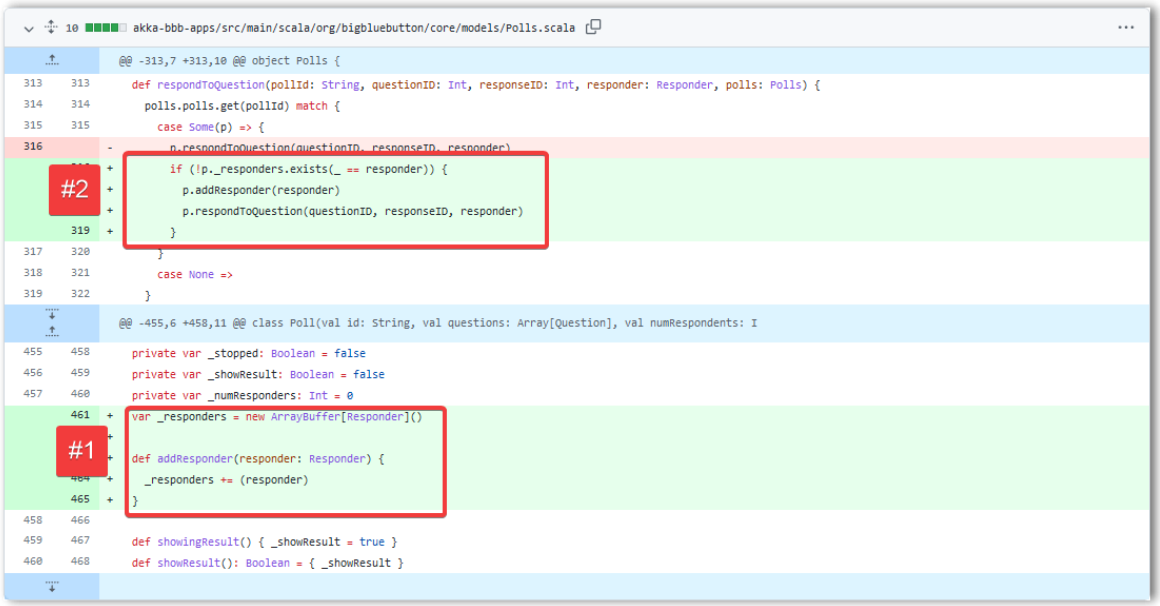
C 1.7.1 Steckbrief

Fix-Bewertung	i.O.
CWE	732 : Incorrect Permission Assignment for Critical Resource
CVE-Kurzbeschreibung	In BigBlueButton before 2.2.29, a user can vote more than once in a single poll.
Veröffentlichung	19.11.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-28953/
CVSS-Score	4.0
Finder-Beschreibung	--

Zusammenfassung	Mehrfach-Voting möglich
Betroffene Versionen	< 2.2.29

C 1.7.2 Korrektur-Informationen

Es wurde eine Liste mit Respondern angelegt (#1), welche bei jedem Vote durchsucht wird, ob der aktuelle Nutzer bereits gevotet hat (#2): https://github.com/bigbluebutton/bigbluebutton/commit/d2cb02b3bd670265c6b1ba003f87fc261e0ac3e1.



Anmerkung: im Screenshot ist nur ein Ausschnitt des Commits dokumentiert, welcher das Vorgehen exemplarisch illustriert.

C 1.7.3 Bewertung

i.O.

C 1.8 CVE-2020-27642

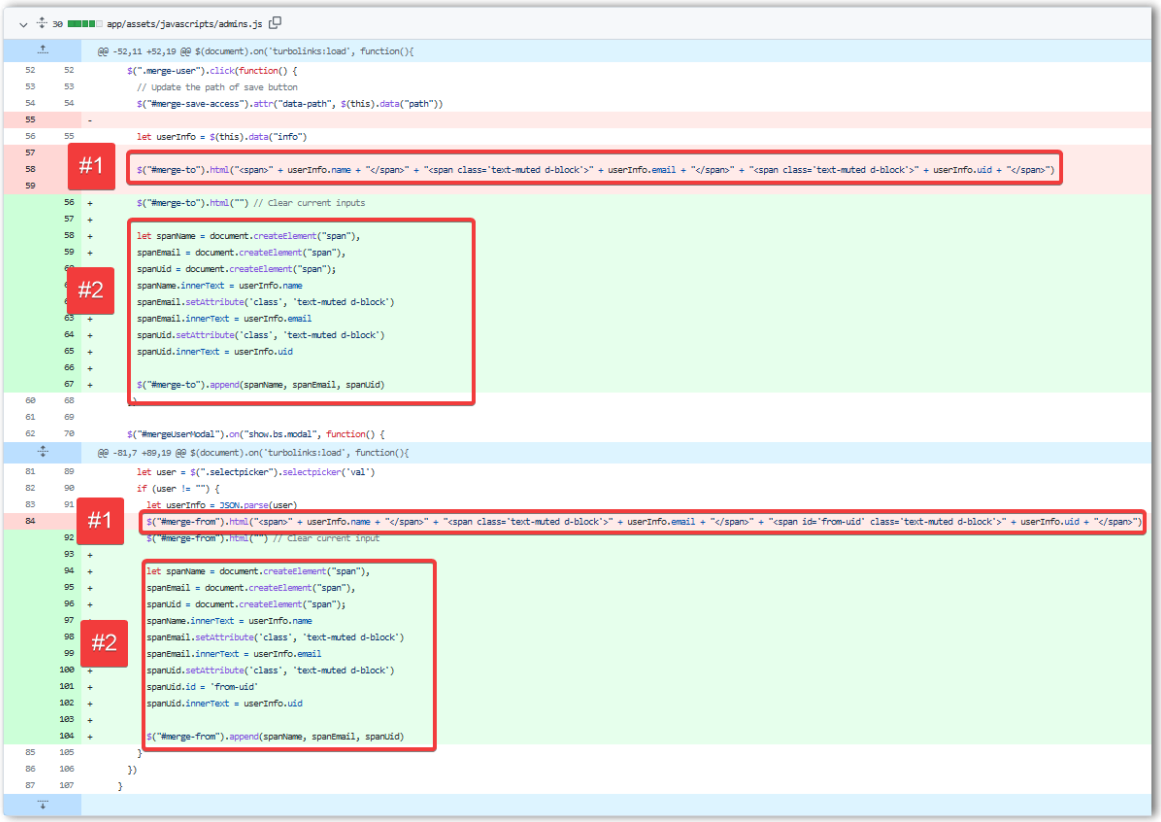
C 1.8.1 Steckbrief

Fix-Bewertung	i.O.
CWE	79 : Failure to Preserve Web Page Structure ('Cross-site Scripting')
CVE-Kurzbeschreibung	A cross-site scripting (XSS) vulnerability exists in the 'merge account' functionality in admins.js in BigBlueButton Green-light 2.7.6.
Veröffentlichung	22.10.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-27642/
CVSS-Score	4.3

Finder-Beschreibung	--
Zusammenfassung	XSS in merge-account-Funktion
Betroffene Versionen	<= 2.7.6

C 1.8.2 Korrektur-Informationen

Im zugehörigen PULL-Request **https://github.com/bigbluebutton/greenlight/pull/2214/files** wird das alte Text-zu-HTML-Vorgehen (es wird vom Browser ein konkatenierter String geparst) an zwei betroffenen Stellen (#1) durch direkte DOM-Manipulation mit sicheren API-Aufrufen (#2) ersetzt:



C 1.8.3 Bewertung

i.O.

C 1.9 CVE-2020-27613

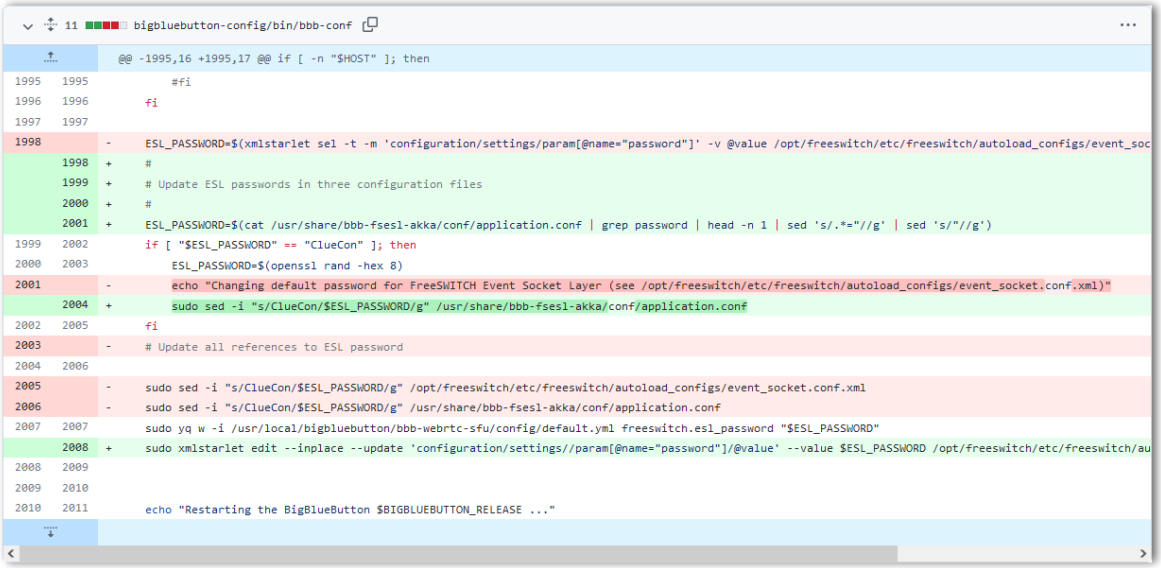
C 1.9.1 Steckbrief

Fix-Bewertung	i.O.
CWE	312 : Cleartext Storage of Sensitive Information

CVE-Kurzbeschreibung	The installation procedure in BigBlueButton before 2.2.28 (or earlier) uses ClueCon as the FreeSWITCH password, which allows local users to achieve unintended FreeSWITCH access.
Veröffentlichung	21.10.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-27613/
CVSS-Score	4.6
Finder-Beschreibung	https://www.golem.de/news/big-blue-button-das-grosse-blaue-sicherheitsrisiko-2010-151610.html
Zusammenfassung	Software Freeswitch wurde standardmäßig mit dem Default-Passwort „ClueCon“ konfiguriert.
Betroffene Versionen	< 2.2.28

C 1.9.2 Korrektur-Informationen

Im zugehörigen Commit https://github.com/bigbluebutton/bigbluebutton/commit/e7d2190f15ba715e56b53433b67e0555a78dcf71 wird der bereits bestehende Mechanismus zur Ersetzung des Standard-Passwortes angepasst:



C 1.9.3 Bewertung

i.O.

C 1.10 CVE-2020-27612

C 1.10.1 Steckbrief

Fix-Bewertung	Teile vom Nutzernamen werden immer noch geleaked.
CWE	200 : Information Exposure

CVE-Kurzbeschreibung	Greenlight in BigBlueButton through 2.2.28 places usernames in room URLs, which may represent an unintended information leak to users in a room, or an information leak to outsiders if any user publishes a screenshot of a browser window.
Veröffentlichung	21.10.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-27612/
CVSS-Score	4.0
Finder-Beschreibung	Lediglich auf BBB-Doc verlinkt: https://docs.bigbluebutton.org/admin/privacy.html
Zusammenfassung	Raum-URL enthält den Nutzernamen.
Betroffene Versionen	<= 2.2.28

C 1.10.2 Korrektur-Informationen

Ein „zugehöriger“ Commit konnte nicht identifiziert werden. Auch heute enthält der Raumname noch immer die ersten 3 Buchstaben des Nutzernamens als Prefix (z.B. `mir-hfc-ygr` für den Nutzer `mirko`)

C 1.10.3 Bewertung

Aus Sicht der Information-Leakage-Problematik ist die Benennung der Räume auch heute noch nicht 100% Privacy-Compliant implementiert.

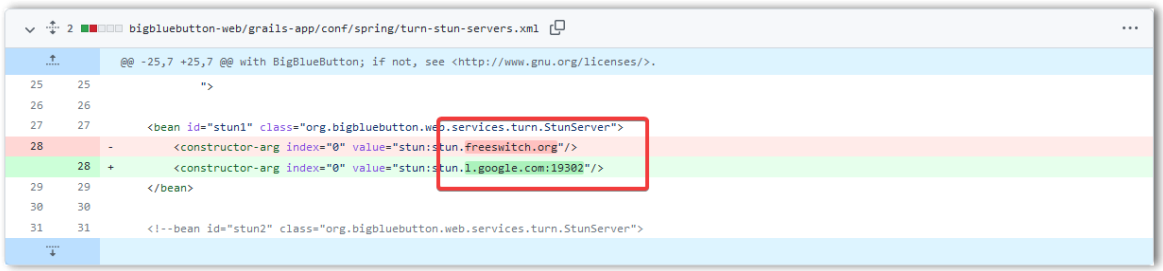
C 1.11 CVE-2020-27611

C 1.11.1 Steckbrief

Fix-Bewertung	STUN-Server von Google kommen zum Einsatz. Aus Privacy-Sicht ist dies mind. “fraglich”.
CWE	327 : Use of a Broken or Risky Cryptographic Algorithm
CVE-Kurzbeschreibung	BigBlueButton through 2.2.28 uses STUN/TURN resources from a third party, which may represent an unintended endpoint.
Veröffentlichung	21.10.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-27611/
CVSS-Score	7.5
Finder-Beschreibung	Lediglich auf BBB-Doc verlinkt: https://docs.bigbluebutton.org/admin/privacy.html
Zusammenfassung	Einsatz des Freeswitch.org-Servers als STUN-Server
Betroffene Versionen	<= 2.2.28

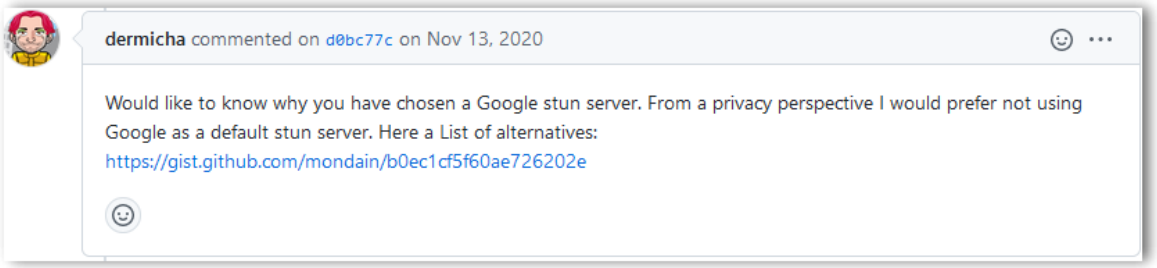
C 1.11.2 Korrektur-Informationen

Es wurde auf den STUN-Server von Google gewechselt (https://github.com/bigbluebutton/bigbluebutton/commit/d0bc77c3dbd858295004f15d7a57ec35e6b203d6):



C 1.11.3 Bewertung

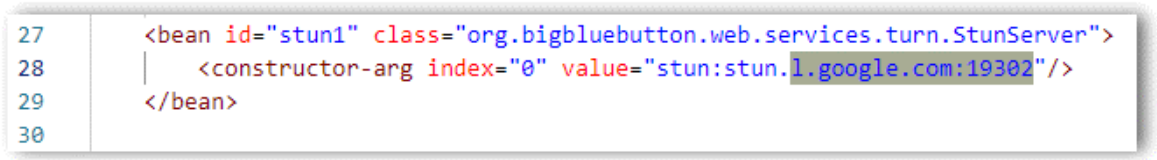
Wie auch einem Kommentar im o.g. Commit zu entnehmen ist:



‚ist es aus Privacy-Sicht mindestens „interessant“, dass man sich hier für Google entschieden hat. Ein möglicher Grund hierfür könnten bekannte Verwundbarkeiten in Freeswitch-Server gewesen sein, jedoch konnte darüber bei unseren Recherchen nichts in Erfahrung gebracht werden.

Hinweis: Auch heute kommt der STUN-Server von Google noch an einigen Stellen zum Einsatz:

`turn-stun-servers.xml`:



`vars.xml`:

```
302  <X-PRE-PROCESS cmd="set" data="external_rtp_ip=stun:stun.l.google.com:19302"/>
303
304  <!-- external_sip_ip
305      Used as the public IP address for SDP.
306       Can be an one of:
307          ip address: "12.34.56.78"
308          a stun server lookup: "stun:stun.server.com"
309          a DNS name: "host:host.server.com"
310       where fs.mydomain.com is a DNS A record-useful when fs is on
311       a dynamic IP address, and uses a dynamic DNS updater.
312       If unspecified, the bind_server_ip value is used.
313       Used by: sofia.conf.xml dingaling.conf.xml
314  -->
315  <X-PRE-PROCESS cmd="set" data="external_sip_ip=stun:stun.l.google.com:19302"/>
```

sip.js:

```
18096  function defaultPeerConnectionConfiguration() {
18097      const configuration = {
18098          bundlePolicy: "balanced",
18099          certificates: undefined,
18100          iceCandidatePoolSize: 0,
18101          iceServers: [{ urls: "stun:stun.l.google.com:19302" }],
18102          iceTransportPolicy: "all",
18103          peerIdentity: undefined,
18104          rtcpMuxPolicy: "require"
18105      };
18106      return configuration;
18107  }
```

C 1.12 CVE-2020-27610

C 1.12.1 Steckbrief

Fix-Bewertung	Es wurde eine ausreichende Cmd-Option eingeführt, welche jedoch das Paradigma „Secure Defaults“ verletzt.
CWE	200 : Information Exposure
CVE-Kurzbeschreibung	The installation procedure in BigBlueButton before 2.2.28 (or earlier) exposes certain network services to external interfaces, and does not automatically set up a firewall configuration to block external access.
Veröffentlichung	21.10.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-27610/
CVSS-Score	5.0
Finder-Beschreibung	https://www.golem.de/news/big-blue-button-das-grosse-blaue-sicherheitsrisiko-2010-151610.html

Zusammenfassung	Bei der Installation wurden standardmäßig Netzwerkservices über extern erreichbare Ports exponiert, die technisch aber nur lokal erreichbar sein müssen.
Betroffene Versionen	< 2.2.28

C 1.12.2 Korrektur-Informationen

Im Commit https://github.com/bigbluebutton/bbb-install/commit/9b98cd4c82a39f36a3df8b9af2416163b1f41002 werden im `bbb-install.sh` Skript UFW-Firewall (UncomplicatedFirewall) Regeln hinzugefügt, welche bei Ausführung des Skriptes mit der `-w` Option den Zugriff auf bestimmte Dienste von außen verhindern.

Hinweis: Ohne die `-w` Option erfolgt keine Aktivierung der UFW-Regeln. Das Skript weist auch nicht darauf hin, falls die Option nicht angegeben wurde. Die Beispielaufrufe auf den Installation-Dokumentationsseiten (z.B. https://github.com/bigbluebutton/bbb-install) weist auf die Option recht prominent hin.

C 1.12.3 Bewertung

Auch, wenn in der Installationsdokumentation an verschiedenen Stellen sehr prominent auf diese Option hingewiesen wird, ist es fraglich, warum nicht umgekehrt herangegangen wurde und die Firewall zum Standard erklärt wurde (Secure Defaults!). Vermutlich wählte man diese Herangehensweise aufgrund von Rückwärtskompatibilitäten und Komfort. Daraus resultierende Probleme, wie z.B. Verlust des Serverzugriffs, weil SSHD nicht auf Port 22 läuft, sind aus unserer Sicht als weniger kritisch, als das ausversehen vergessen der `-w` Option bei der Installation einzuschätzen.

C 1.13 CVE-2020-27609

C 1.13.1 Steckbrief

Fix-Bewertung	nicht optimal (Privacy!)
CWE	CWE-863: Incorrect Authorization
CVE-Kurzbeschreibung	BigBlueButton through 2.2.28 records a video meeting despite the deactivation of video recording in the user interface. This may result in data storage beyond what is authorized for a specific meeting topic or participant.
Veröffentlichung	21.10.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-27609/
CVSS-Score	5.0
Finder-Beschreibung	https://www.golem.de/news/big-blue-button-das-grosse-blaue-sicherheitsrisiko-2010-151610.html PLUS Verlinkung auf BBB-Doc: https://docs.bigbluebutton.org/admin/privacy.html

Zusammenfassung	Die Aufzeichnung der Session startet und stoppt nicht durch Betätigung des Buttons im Frontend und ist standardmäßig aktiv.
Betroffene Versionen	<= 2.2.28

C 1.13.2 Korrektur-Informationen

Dieses Verhalten wurde im BBB-Team an diversen Stellen diskutiert (z.B. https://github.com/bigbluebutton/greenlight/issues/1163) und final wurde entschieden, dass das in der CVE angemahnte Verhalten nicht grundlegend korrigiert wird. Es wurde lediglich die Möglichkeit geschaffen, das Recording beim Erstellen eines Raumes (über Greenlight UI) zu aktivieren oder zu deaktivieren (https://github.com/bigbluebutton/greenlight/pull/1296).

Hinweis: Mit der Greenlight-Version 2.7-alpha wurde über den PULL-Request https://github.com/bigbluebutton/greenlight/pull/1951/files#diff-c1fd91cb1911a0512578b99f657554526f3e1421decdb9e908712beab57e10f9 das Standard-Verhalten so angepasst, dass die Aufnahme automatisch gestartet wird und standardmäßig bei der Erstellung eines Raumes keine Auswahl für „Recording ja/nein“ (in eine RAW-Datei) mehr existiert. Möchte man diese Auswahlmöglichkeit bei der Raumanlage, wieder aktivieren, muss die Konfigurationsdatei `config/application.rb` folgendermaße angepasst werden: `config.require_consent_default = "true"`. Dies ist auch über die Site-Administrationsoberfläche möglich.

Der start/stop-Button in der UI ist lediglich zur Protokollierung des gewünschten Start- und Endzeitpunktes für die spätere Video-Erzeugung aus der RAW-Datei relevant. Die Aufzeichnung läuft (falls aktiviert) permanent.

Ungenutzte Aufzeichnungen werden standardmäßig nach 1 Woche gelöscht.

C 1.13.3 Bewertung

Auch, wenn vielen der Entscheidungen, welche zu Code-Anpassungen im Recording führten, eine fachlich nachvollziehbare Argumentation zugrundeliegt, ist das umgesetzte Verhalten (insbesondere die Anpassung mit Greenlight-Version 2.7-alpha) aus Privacy-Sicht nicht ideal umgesetzt.

Insbesondere auch die heute aktive Standard-Grundeinstellung wirft Fragen auf!

C 1.14 CVE-2020-27608

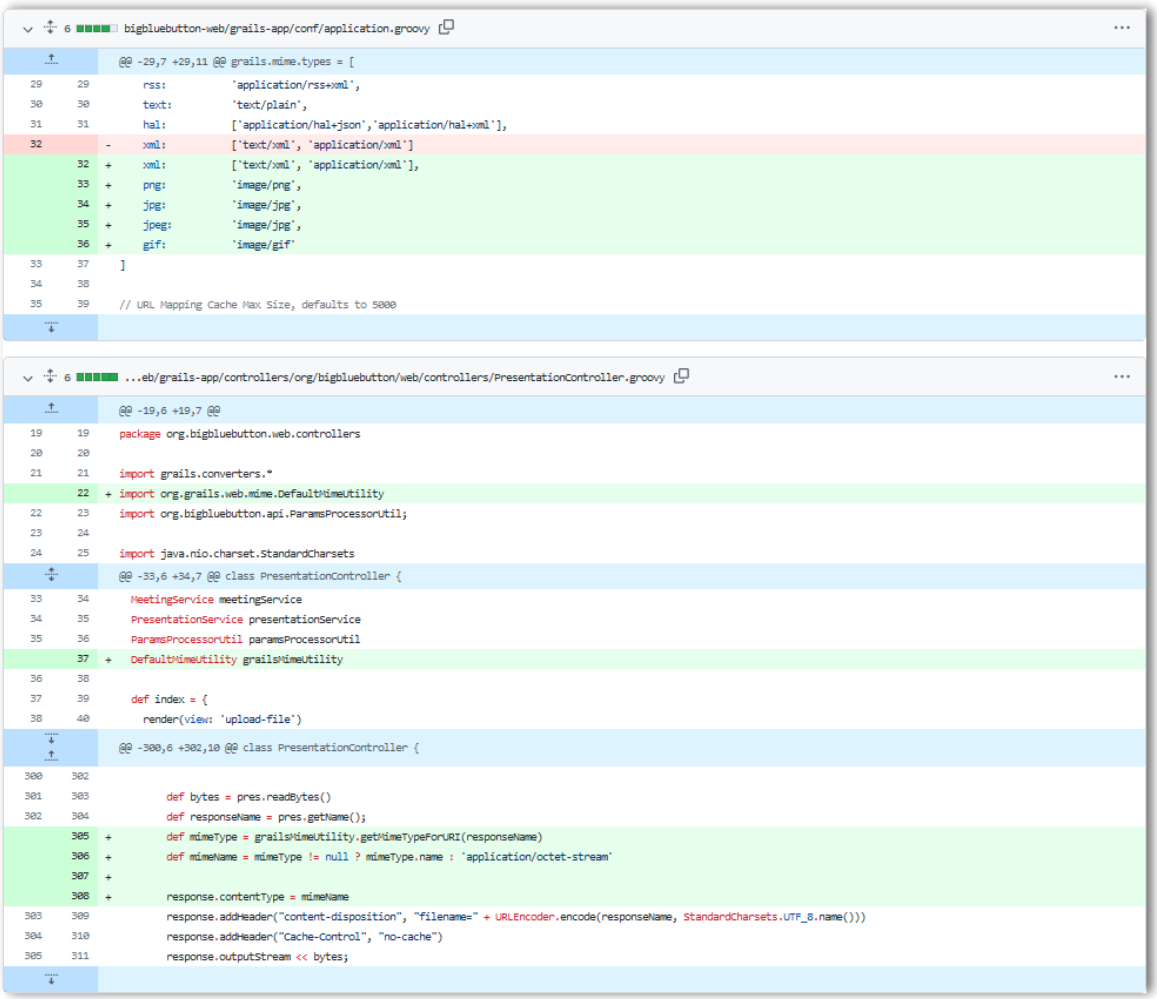
C 1.14.1 Steckbrief

Fix-Bewertung	i.O.
CWE	79 : Failure to Preserve Web Page Structure ('Cross-site Scripting')

CVE-Kurzbeschreibung	In BigBlueButton before 2.2.28 (or earlier), uploaded presentations are sent to clients without a Content-Type header, which allows XSS, as demonstrated by a .png file extension for an HTML document.
Veröffentlichung	21.10.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-27608/
CVSS-Score	4.3
Finder-Beschreibung	https://www.golem.de/news/big-blue-button-das-grosse-blaue-sicherheitsrisiko-2010-151610.html
Zusammenfassung	Hochgeladene Präsentationen werden ohne Mime-Type wieder heruntergeladen, wodurch die Möglichkeit von XSS-Vektoren besteht.
Betroffene Versionen	< 2.2.28

C 1.14.2 Korrektur-Informationen

Im Commit
https://github.com/bigbluebutton/bigbluebutton/commit/79361bd4859145f888a88d59d92b1a83ccc8ab23 (vom 16.04.2020) wird der `Content-Type` Header eingefügt und weitere Mime-Types konfiguriert.



```
bigbluebutton-web/grails-app/conf/application.groovy
@@ -29,7 +29,11 @@ grails.mime.types = [
    rss:          'application/rss+xml',
    text:         'text/plain',
    hal:          ['application/hal+json','application/hal+xml'],
-   xml:          ['text/xml', 'application/xml']
+   xml:          ['text/xml', 'application/xml'],
+   png:          'image/png',
+   jpg:          'image/jpg',
+   jpeg:         'image/jpg',
+   gif:          'image/gif'
]

// URL Mapping Cache Max Size, defaults to 5000

...eb/grails-app/controllers/org/bigbluebutton/web/controllers/PresentationController.groovy
@@ -19,6 +19,7 @@
package org.bigbluebutton.web.controllers

import grails.converters.*
+import org.grails.web.mime.DefaultMimeUtility
import org.bigbluebutton.api.ParamsProcessorUtil;

import java.nio.charset.StandardCharsets
@@ -33,6 +34,7 @@ class PresentationController {
  MeetingService meetingService
  PresentationService presentationService
  ParamsProcessorUtil paramsProcessorUtil
+ DefaultMimeUtility grailsMimeUtility

  def index = {
    render(view: 'upload-file')
@@ -300,6 +302,10 @@ class PresentationController {

        def bytes = pres.readBytes()
        def responseName = pres.getName();
+       def mimeType = grailsMimeUtility.getMimeTypeForURI(responseName)
+       def mimeName = mimeType != null ? mimeType.name : 'application/octet-stream'
+
+       response.contentType = mimeName
        response.addHeader("content-disposition", "filename=" + URLEncoder.encode(responseName, StandardCharsets.UTF_8.name()))
        response.addHeader("Cache-Control", "no-cache")
        response.outputStream << bytes;
```

C 1.14.3 Bewertung

i.O.

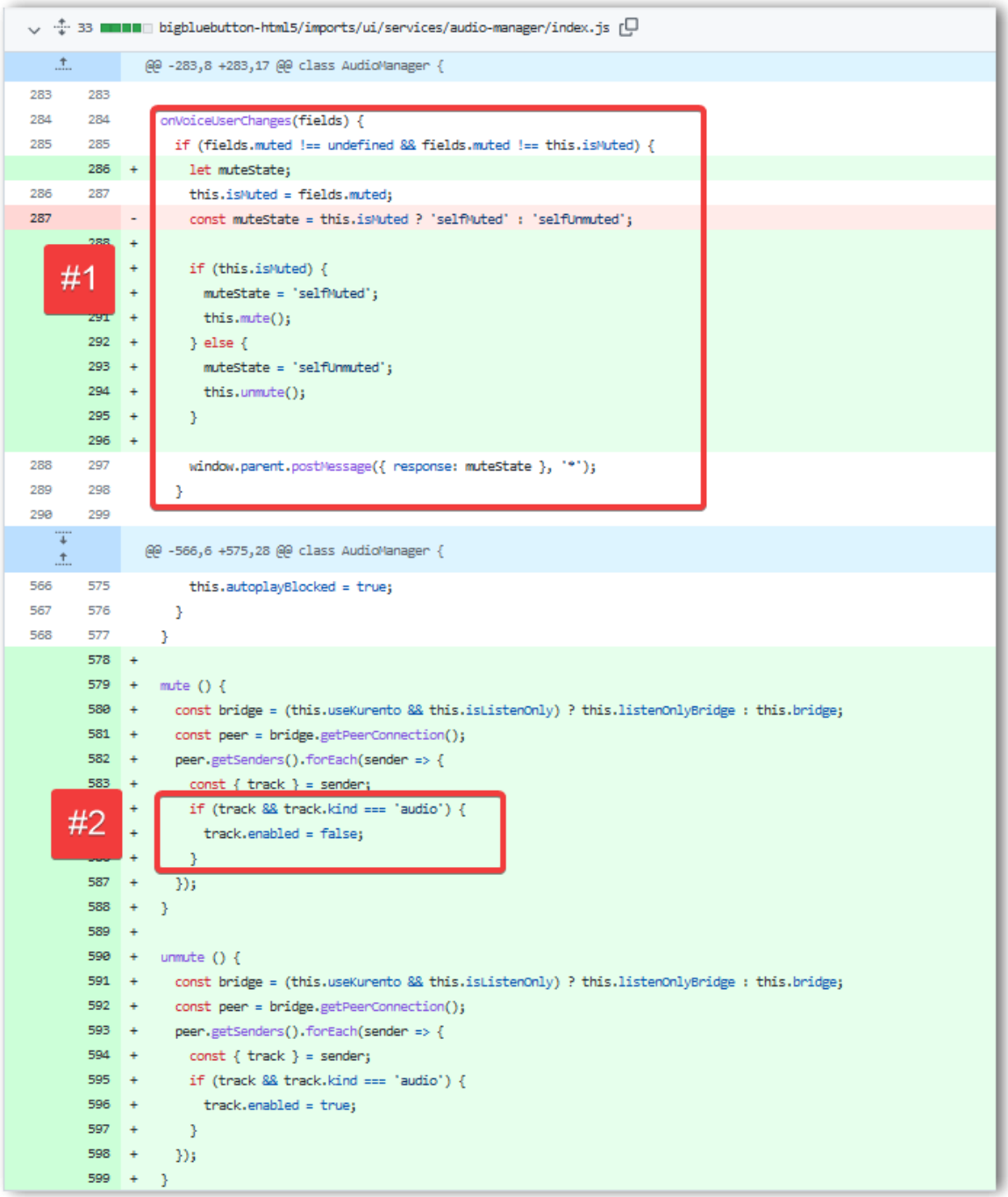
C 1.15 CVE-2020-27607

C 1.15.1 Steckbrief

Fix-Bewertung	i.O.
CWE	--
CVE-Kurzbeschreibung	In BigBlueButton before 2.2.28 (or earlier), the client-side Mute button only signifies that the server should stop accepting audio data from the client. It does not directly configure the client to stop sending audio data to the server, and thus a modified server could store the audio data and/or transmit it to one or more meeting participants or other third parties.
Veröffentlichung	21.10.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-27607/
CVSS-Score	6.4
Finder-Beschreibung	https://www.golem.de/news/big-blue-button-das-grosse-blaue-sicherheitsrisiko-2010-151610.html
Zusammenfassung	Audiodaten werden weiter an den Server gesendet, auch, wenn der Nutzer seinen Client gemutet hat.
Betroffene Versionen	< 2.2.28

C 1.15.2 Korrektur-Informationen

Diese CVE wird im PULL-Request
https://github.com/bigbluebutton/bigbluebutton/pull/10423 behandelt:



Im Fall eines Zustandwechsels (#1) wird über die `mute()`-Funktion die zugehörige Übertragung deaktiviert (#2).

Schaut man sich die laufende Anwendung an, stellt man fest, dass weitere Audio-Daten übertragen werden. Dabei handelt es sich jedoch um sogenannte „silent frames“, welche nur gesendet werden, um keinen erneuten Verbindungsaufbau mit dem FreeSWITCH-Endpunkt durchführen zu müssen.

C 1.15.3 Bewertung

i.O.

Hinweis: Grundsätzlich wäre es zur Ausnutzung dieser Schwachstelle auch erforderlich, dass ein Angreifer Zugriff auf den Server haben müsste (z.B. Rolle admin). Dadurch wäre er aber auch in der Lage, einen modifizierten Client (mit ausgebauter Abschaltung) an die Nutzer auszuliefern.

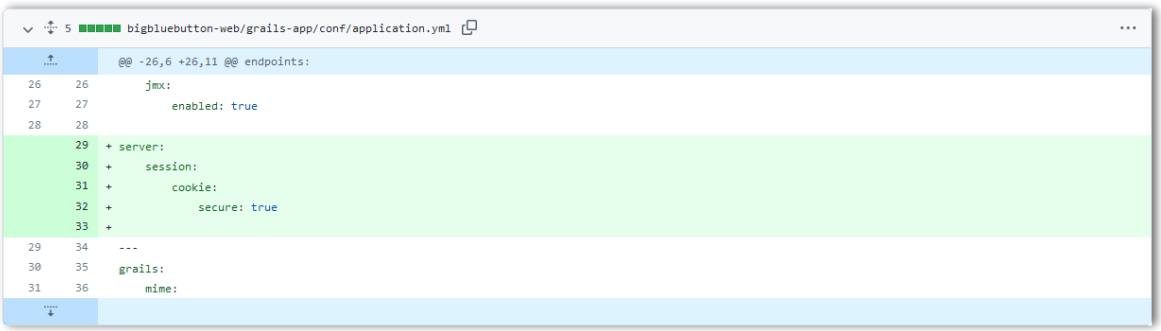
C 1.16 CVE-2020-27606

C 1.16.1 Steckbrief

Fix-Bewertung	i.O.
CWE	--
CVE-Kurzbeschreibung	BigBlueButton before 2.2.28 (or earlier) does not set the secure flag for the session cookie in an https session, which makes it easier for remote attackers to capture this cookie by intercepting its transmission within an http session.
Veröffentlichung	21.10.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-27606/
CVSS-Score	5.0
Finder-Beschreibung	https://www.golem.de/news/big-blue-button-das-grosse-blaue-sicherheitsrisiko-2010-151610.html
Zusammenfassung	`secure`-Flag fehlt am Session-Cookie
Betroffene Versionen	< 2.2.28

C 1.16.2 Korrektur-Informationen

Im Commit https://github.com/bigbluebutton/bigbluebutton/commit/a98c4b68b50accebb7bcc589feb0bc4305c1672a wird das `secure`-Flag gesetzt.



C 1.16.3 Bewertung

i.O.

C 1.17 CVE-2020-27605

C 1.17.1 Steckbrief

Fix-Bewertung	i.O.
CWE	--
CVE-Kurzbeschreibung	BigBlueButton through 2.2.28 uses Ghostscript for processing of uploaded EPS documents, and consequently may be subject to attacks related to a "schwache Sandbox."
Veröffentlichung	21.10.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-27605/
CVSS-Score	7.5
Finder-Beschreibung	https://www.golem.de/news/big-blue-button-das-grosse-blaue-sicherheitsrisiko-2010-151610.html
Zusammenfassung	Erfolgreiche Angriffe auf die Ghostscript-/LibreOffice-Verarbeitung ermöglichen direkten Zugriff auf den Server (bbb-web) z.B. unter Ausnutzung von CVE-2020-27603 in C1.19.
Betroffene Versionen	<= 2.2.28

C 1.17.2 Korrektur-Informationen

Mit dem PULL-Request https://github.com/bigbluebutton/bigbluebutton/pull/9977 wird die Ghostscript-Verarbeitung in einen Docker-Container ausgelagert.

C 1.17.3 Bewertung

i.O.

C 1.18 CVE-2020-27604

C 1.18.1 Steckbrief

Fix-Bewertung	i.O.
CWE	116 : Improper Encoding or Escaping of Output
CVE-Kurzbeschreibung	BigBlueButton before 2.3 does not implement LibreOffice sandboxing. This might make it easier for remote authenticated users to read the API shared secret in the bigbluebutton.properties file. With the API shared secret, an attacker can (for example) use api/join to join an arbitrary meeting regardless of its guestPolicy setting.
Veröffentlichung	21.10.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-27604/
CVSS-Score	4.0

Finder-Beschreibung	https://www.golem.de/news/big-blue-button-das-grosse-blaue-sicherheitsrisiko-2010-151610.html PLUS Verlinkung auf BBB-Doc: https://docs.bigbluebutton.org/dev/api.html
Zusammenfassung	Erfolgreiche Angriffe auf die Ghostscript-/LibreOffice-Verarbeitung ermöglichen direkten Zugriff auf den Server (bbb-web) z.B. unter Ausnutzung von CVE-2020-27603 in C1.19.
Betroffene Versionen	<= 2.3

C 1.18.2 Korrektur-Informationen

Durch die mit dem PULL-Request https://github.com/bigbluebutton/bigbluebutton/pull/9977 umgesetzte Auslagerung in einen Docker-Container (siehe auch CVE-2020-27605 in C1.17) sowie die Beschränkung der Möglichkeiten des `JODConverter`s im PULL-Request https://github.com/bigbluebutton/bigbluebutton/pull/10619 (siehe auch CVE-2020-27603 in C1.19) wird diese CVE ebenfalls addressiert.

C 1.18.3 Bewertung

i.O.

C 1.19 CVE-2020-27603

C 1.19.1 Steckbrief

Fix-Bewertung	i.O.
CWE	--
CVE-Kurzbeschreibung	BigBlueButton before 2.2.27 has an unsafe JODConverter setting in which LibreOffice document conversions can access external files.
Veröffentlichung	21.10.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-27603/
CVSS-Score	5.0
Finder-Beschreibung	https://www.golem.de/news/big-blue-button-das-grosse-blaue-sicherheitsrisiko-2010-151610.html
Zusammenfassung	Zugriff auf externe Dateien durch speziell angepasste Dokumente bei der LibreOffice-Verarbeitung möglich.
Betroffene Versionen	<= 2.2.27

C 1.19.2 Korrektur-Informationen

Im PULL-Request https://github.com/bigbluebutton/bigbluebutton/pull/10619 **wurde der** `JODConverter` **entsprechend konfiguriert:**

```
25 ...on-web/src/main/java/org/bigbluebutton/presentation/imp/OfficeToPdfConversionService.java

41  36      private OfficeDocumentValidator2 officeDocumentValidator;
42  37      private final OfficeManager officeManager;
43  38      private final LocalConverter documentConverter;
44  39      private boolean skipOfficePrecheck = false;
45          -
46  40      public OfficeToPdfConversionService() throws OfficeException {
        41  +     final Map<String, Object> loadProperties = new HashMap<>();
        42  +     loadProperties.put("Hidden", true);
        43  +     loadProperties.put("ReadOnly", true);
        44  +     loadProperties.put("UpdateDocMode", UpdateDocMode.NO_UPDATE);
47  45          officeManager = localOfficeManager
48  46            .builder()
49  47            .portNumbers(8100, 8101, 8102, 8103, 8104)
50  48            .build();
51  49          documentConverter = LocalConverter
52  50            .builder()
53  51            .officeManager(officeManager)
        52  +     .loadProperties(loadProperties)
54  53            .filterChain(new OfficeDocumentConversionFilter())
55  54            .build();
56  55      }
57          -
58  56      /*
59  57       * Convert the Office document to PDF. If successful, update
60  58       * UploadPresentation.uploadedFile with the new PDF out and

@@ -74,7 +72,6 @@ public UploadedPresentation convertOfficeToPdf(UploadedPresentation pres) {
```

C 1.19.3 Bewertung

i.O.

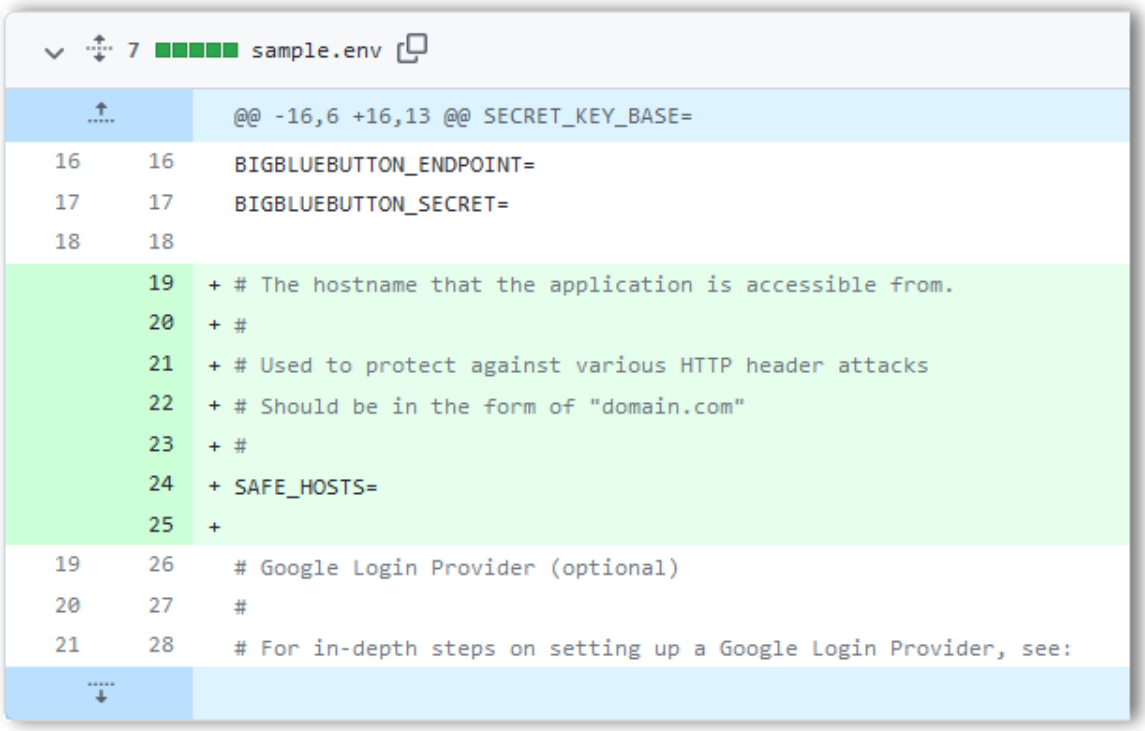
C 1.20 CVE-2020-26163

C 1.20.1 Steckbrief

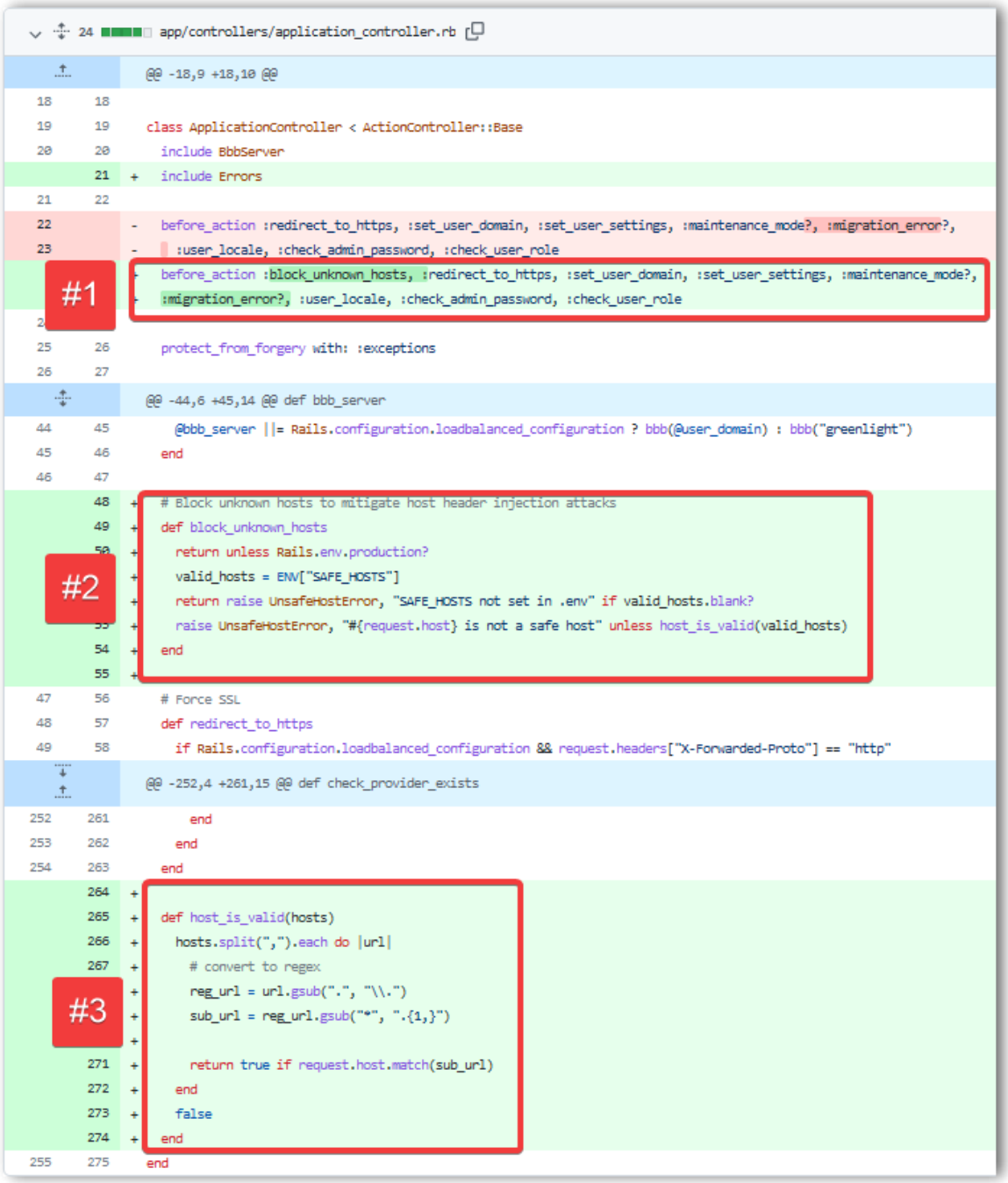
Fix-Bewertung	i.O.
CWE	--
CVE-Kurzbeschreibung	BigBlueButton Greenlight before 2.5.6 allows HTTP header (Host and Origin) attacks, which can result in Account Take-over if a victim follows a spoofed password-reset link.
Veröffentlichung	30.09.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-26163/
CVSS-Score	6.8
Finder-Beschreibung	https://www.sakshamanand.com/blog/host-header-injection-bigbluebutton
Zusammenfassung	Anpassung von HOST- oder ORIGIN-Header führt zur Generierung falscher (Angreifer-kontrollierter) URLs in Mails für die Passwort-Reset-Funktion. Reset-Token-Diebstahl und Phishing-Angriffe werden dadurch ermöglicht.
Betroffene Versionen	< 2.5.6

C 1.20.2 Korrektur-Informationen

Im PULL-Request https://github.com/bigbluebutton/greenlight/pull/1543/files wird die Konfiguration um die Möglichkeit der Spezifikation sicherer Hosts erweitert:



Im `application_controller.rb` wird die `before_action` um den Aufruf `:block_unknown_hosts` erweitert (#1), welche den aktuellen Host mit der o.g. `SAFE_HOSTS` Umgebungsvariable vergleicht und prüft, ob dieser dort gelistet ist (#2 bzw. #3). Mehrfachnennungen sind mit Komma getrennt möglich und es werden reguläre Audrücke unterstützt. Punkte werden literalisiert:




```
24 app/controllers/application_controller.rb

@@ -18,9 +18,10 @@
18 18
19 19  class ApplicationController < ActionController::Base
20 20    include BbbServer
21 +    include Errors
21 22
22 -    before_action :redirect_to_https, :set_user_domain, :set_user_settings, :maintenance_mode?, :migration_error?,
23 -      :user_locale, :check_admin_password, :check_user_role
+    before_action :block_unknown_hosts, :redirect_to_https, :set_user_domain, :set_user_settings, :maintenance_mode?,
+    :migration_error?, :user_locale, :check_admin_password, :check_user_role
25 26    protect_from_forgery with: :exceptions
26 27
@@ -44,6 +45,14 @@ def bbb_server
44 45      @bbb_server ||= Rails.configuration.loadbalanced_configuration ? bbb(@user_domain) : bbb("greenlight")
45 46    end
46 47
+    # Block unknown hosts to mitigate host header injection attacks
+    def block_unknown_hosts
+      return unless Rails.env.production?
+      valid_hosts = ENV["SAFE_HOSTS"]
+      return raise UnsafeHostError, "SAFE_HOSTS not set in .env" if valid_hosts.blank?
+      raise UnsafeHostError, "#{request.host} is not a safe host" unless host_is_valid(valid_hosts)
+    end
+
47 56    # Force SSL
48 57    def redirect_to_https
49 58      if Rails.configuration.loadbalanced_configuration && request.headers["X-Forwarded-Proto"] == "http"
@@ -252,4 +261,15 @@ def check_provider_exists
252 261        end
253 262      end
254 263    end
+
+    def host_is_valid(hosts)
+      hosts.split(",").each do |url|
+        # convert to regex
+        reg_url = url.gsub(".", "\\.")
+        sub_url = reg_url.gsub("*", ".{1,}")
+
+        return true if request.host.match(sub_url)
+      end
+      false
+    end
255 275  end
```

C 1.20.3 Bewertung

i.O.

C 1.21 CVE-2020-25820

C 1.21.1 Steckbrief

Fix-Bewertung	i.O.
CWE	CWE-918: Server-Side Request Forgery (SSRF)

CVE-Kurzbeschreibung	BigBlueButton before 2.2.7 allows remote authenticated users to read local files and conduct SSRF attacks via an uploaded Office document that has a crafted URL in an ODF xlink field.
Veröffentlichung	21.10.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-25820/
CVSS-Score	4.0
Finder-Beschreibung	https://www.redteam-pentesting.de/en/advisories/rt-sa-2020-005/-arbitrary-file-disclosure-and-server-side-request-forgery-in-bigbluebutton und https://packetstormsecurity.com/files/159667/BigBlueButton-2.2.25-File-Disclosure-Server-Side-Request-Forgery.html
Zusammenfassung	File-Upload kann für SSRF bzw. zum Auslesen serverseitiger Dateien etc. verwendet werden.
Betroffene Versionen	< 2.2.27

C 1.21.2 Korrektur-Informationen

Im Commit
https://github.com/bigbluebutton/bigbluebutton/commit/71fe1eac1e5bd73a2cd44bd79c001086b250e435 wird der Dokumenten-Konverter mit zusätzlichen Properties konfiguriert:

```
35      -
        32  + import com.sun.star.document.UpdateDocMode;
36      33    import com.google.gson.Gson;
37      -
38      34    public class OfficeToPdfConversionService {
39      35      private static Logger log = LoggerFactory.getLogger(OfficeToPdfConversionService.class);
40      -
41      36      private OfficeDocumentValidator2 officeDocumentValidator;
42      37      private final OfficeManager officeManager;
43      38      private final LocalConverter documentConverter;
44      39      private boolean skipOfficePrecheck = false;
45      -
46      40      public OfficeToPdfConversionService() throws OfficeException {
        41  +     final Map<String, Object> loadProperties = new HashMap<>();
        42  +     loadProperties.put("Hidden", true);
        43  +     loadProperties.put("ReadOnly", true);
        44  +     loadProperties.put("UpdateDocMode", UpdateDocMode.NO_UPDATE);
47      45        officeManager = localOfficeManager
48      46          .builder()
49      47          .portNumbers(8100, 8101, 8102, 8103, 8104)
50      48          .build();
51      49        documentConverter = LocalConverter
52      50          .builder()
53      51          .officeManager(officeManager)
        52  +       .loadProperties(loadProperties)
54      53          .filterChain(new OfficeDocumentConversionFilter())
55      54          .build();
56      55      }
```

Siehe auch CVE-2020-27603 in C1.19.

mgm security partners GmbH – 25.04.2022

C 1.21.3 Bewertung

i.O.

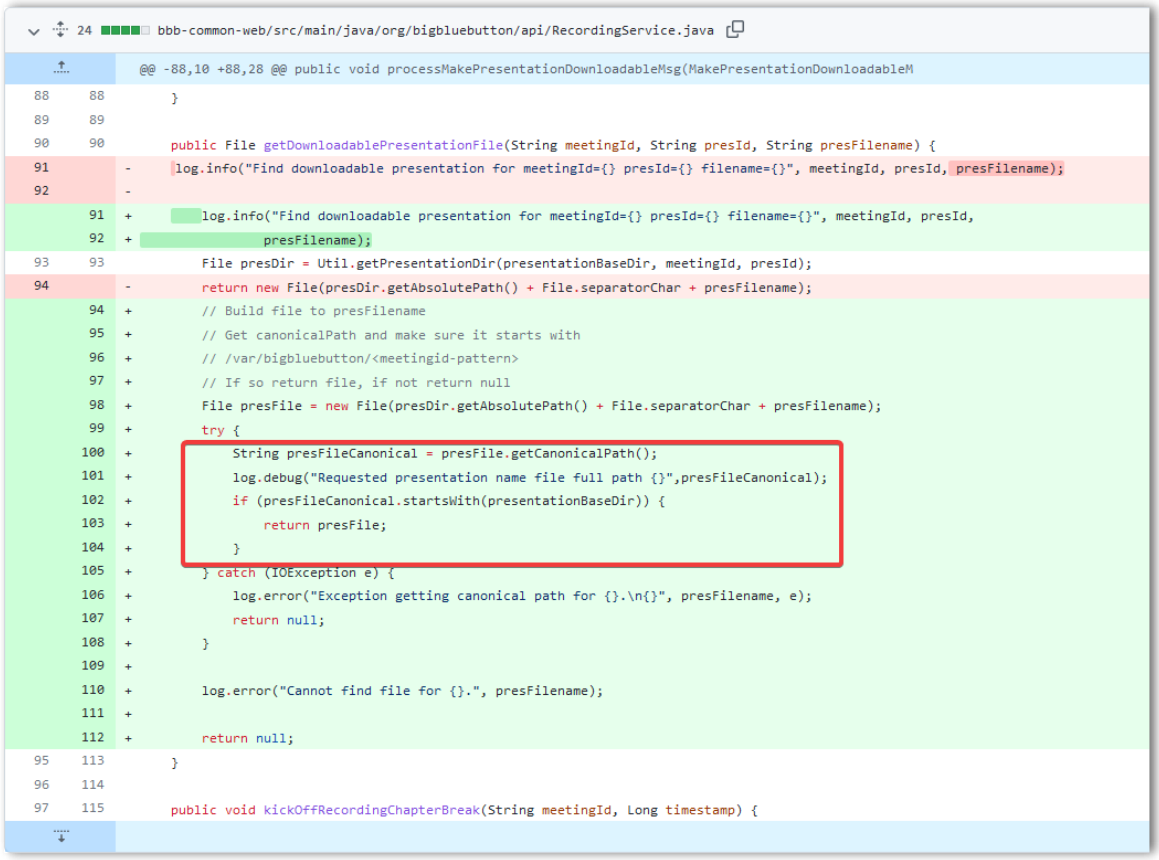
C 1.22 CVE-2020-12443

C 1.22.1 Steckbrief

Fix-Bewertung	i.O.
CWE	22 : Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
CVE-Kurzbeschreibung	BigBlueButton before 2.2.6 allows remote attackers to read arbitrary files because the presfilename (lowercase) value can be a .pdf filename while the presFilename (mixed case) value has a ../ sequence. This can be leveraged for privilege escalation via a directory traversal to bigbluebutton.properties. NOTE: this issue exists because of an ineffective mitigation to CVE-2020-12112 in which there was an attempted fix within an NGINX configuration file, without considering that the relevant part of NGINX is case-insensitive.
Veröffentlichung	29.04.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-12443/
CVSS-Score	7.5
Finder-Beschreibung	https://github.com/mclab-hbrs/BBB-POC
Zusammenfassung	Path-Traversal mit Auslesen von beliebigen Dateien möglich.
Betroffene Versionen	< 2.2.26

C 1.22.2 Korrektur-Informationen

Im zugehörigen Commit
https://github.com/bigbluebutton/bigbluebutton/pull/9259/commits/b21ca8355a57286a1e6df96984b3a4c57679a463 wurde der direkte Datei-Zugriff durch eine Kanonisierung des errechneten Pfades mit einer folgenden Prüfung auf das richtige Meeting-Ordner-Prefix ersetzt:



```
24 bbb-common-web/src/main/java/org/bigbluebutton/api/RecordingService.java

@@ -88,10 +88,28 @@ public void processMakePresentationDownloadableMsg(MakePresentationDownloadableM
88  88      }
89  89
90  90      public File getDownloadablePresentationFile(String meetingId, String presId, String presFilename) {
91      -       log.info("Find downloadable presentation for meetingId={} presId={} filename={}", meetingId, presId, presFilename);
92      -
    91  +       log.info("Find downloadable presentation for meetingId={} presId={} filename={}", meetingId, presId,
    92  +               presFilename);
93  93          File presDir = Util.getPresentationDir(presentationBaseDir, meetingId, presId);
94      -       return new File(presDir.getAbsolutePath() + File.separatorChar + presFilename);
    94  +       // Build file to presFilename
    95  +       // Get canonicalPath and make sure it starts with
    96  +       // /var/bigbluebutton/<meetingid-pattern>
    97  +       // If so return file, if not return null
    98  +       File presFile = new File(presDir.getAbsolutePath() + File.separatorChar + presFilename);
    99  +       try {
   100  +           String presFileCanonical = presFile.getCanonicalPath();
   101  +           log.debug("Requested presentation name file full path {}",presFileCanonical);
   102  +           if (presFileCanonical.startsWith(presentationBaseDir)) {
   103  +               return presFile;
   104  +           }
   105  +       } catch (IOException e) {
   106  +           log.error("Exception getting canonical path for {}.\n{}", presFilename, e);
   107  +           return null;
   108  +       }
   109  +
   110  +       log.error("Cannot find file for {}.", presFilename);
   111  +
   112  +       return null;
95  113      }
96  114
97  115      public void kickOffRecordingChapterBreak(String meetingId, Long timestamp) {
```

C 1.22.3 Bewertung

i.O.

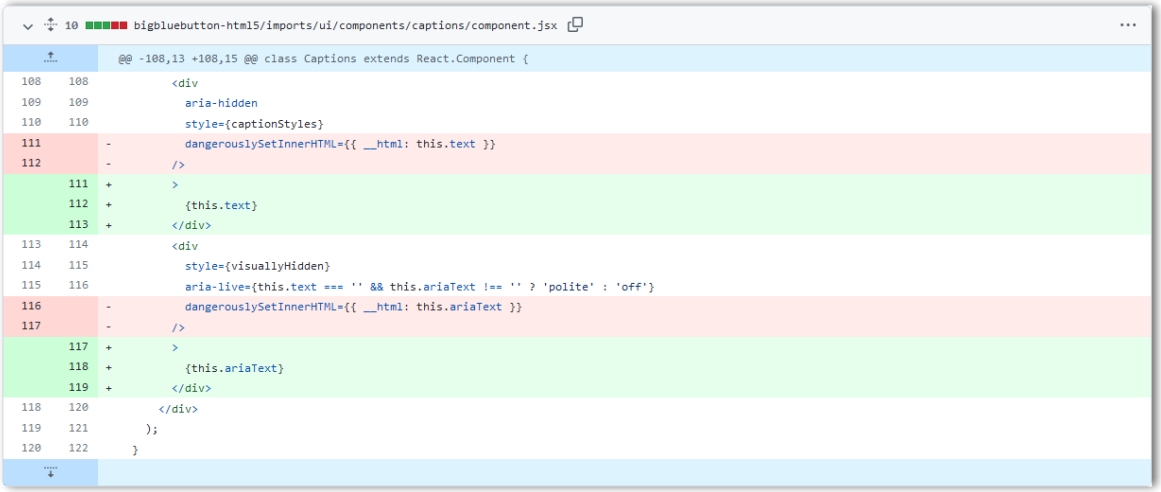
C 1.23 CVE-2020-12113

C 1.23.1 Steckbrief

Fix-Bewertung	i.O.
CWE	79 : Failure to Preserve Web Page Structure ('Cross-site Scripting')
CVE-Kurzbeschreibung	BigBlueButton before 2.2.4 allows XSS via closed captions because `dangerouslySetInnerHTML` in React is used.
Veröffentlichung	23.04.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-12113/
CVSS-Score	4.3
Finder-Beschreibung	https://www.sakshamanand.com/blog/cve-2020-12113/
Zusammenfassung	CrossSiteScripting über React (via `dangerouslySetInnerHTML`) möglich.
Betroffene Versionen	< 2.2.24

C 1.23.2 Korrektur-Informationen

Im zugehörigen Commit https://github.com/bigbluebutton/bigbluebutton/pull/9017/commits/3a6260f6f3d5e3d711d1a86473f601d2e725aacd werden die entsprechenden `dangerouslySetInnerHTML` Verwendungen korrigiert:



C 1.23.3 Bewertung

i.O.

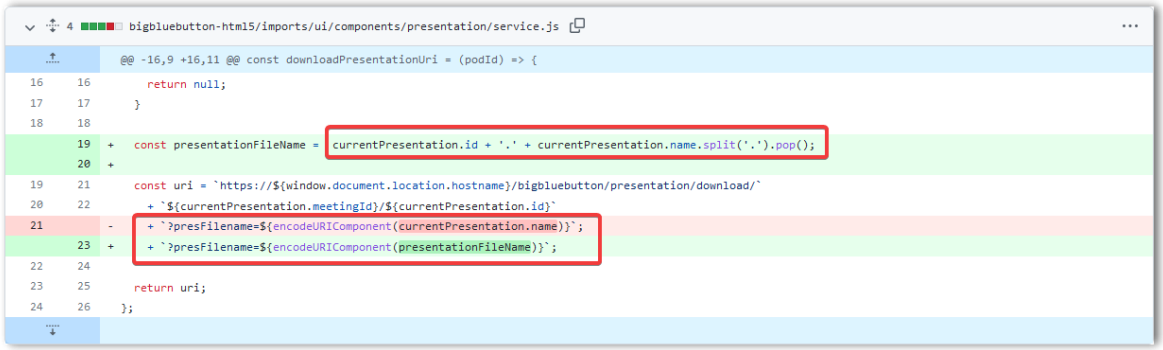
C 1.24 CVE-2020-12112

C 1.24.1 Steckbrief

Fix-Bewertung	i.O.
CWE	200 : Information Exposure
CVE-Kurzbeschreibung	BigBlueButton before 2.2.5 allows remote attackers to obtain sensitive files via Local File Inclusion.
Veröffentlichung	23.04.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-12112/
CVSS-Score	5.0
Finder-Beschreibung	https://twitter.com/thibeault_chenu/status/1249976515917422593
Zusammenfassung	Lokale Datei-Einbindung (dadurch Auslesen sensibler lokaler Dateien) möglich.
Betroffene Versionen	< 2.2.4

C 1.24.2 Korrektur-Informationen

Der zugehörige Commit https://github.com/bigbluebutton/bigbluebutton/commit/535f2d665af254805a28d680ec7dc52e9e358281 korrigiert dies, indem er die ID statt des Namens zum Download der Präsentation verwendet:



C 1.24.3 Bewertung

i.O.

C 2 Jitsi

Stand 17.03.2022 (Übersicht: https://www.cvedetails.com/vendor/16088/Jitsi.html).

C 2.1 CVE-2021-26812 (Out-of-scope)

C 2.1.1 Steckbrief

Fix-Bewertung	Out-of-scope
CWE	79 : Failure to Preserve Web Page Structure ('Cross-site Scripting')
CVE-Kurzbeschreibung	Cross Site Scripting (XSS) in the Jitsi Meet 2.7 through 2.8.3 plugin for Moodle via the "sessionpriv.php" module. This allows attackers to craft a malicious URL, which when clicked on by users, can inject javascript code to be run by the application.
Veröffentlichung	14.04.2021
CVE-Details	https://www.cvedetails.com/cve/CVE-2021-26812/
CVSS-Score	4.3
Finder-Beschreibung	https://github.com/udima-university/moodle-mod_jitsi/files/5893715/Moodle.Jitsi.Plugin.XSS.POC.pdf
Zusammenfassung	XSS in Mod
Betroffene Versionen	>= 2.7 && <= 2.8.3

C 2.1.2 Korrektur-Informationen

Commit https://github.com/udima-university/moodle-mod_jitsi/commit/4364ddbdc2e9ffe6c2bed8db4838936bd70742fc.

C 2.1.3 Bewertung

Nicht Teil von zu untersuchenden Jitsi-Bestandteilen.

C 2.2 CVE-2020-25019 (Out-of-scope)

C 2.2.1 Steckbrief

Fix-Bewertung	Out-of-scope
CWE	345 : Insufficient Verification of Data Authenticity
CVE-Kurzbeschreibung	jitsi-meet-electron (aka Jitsi Meet Electron) before 2.3.0 calls the Electron shell.openExternal function without verifying that the URL is for an http or https resource, in some circumstances.

Veröffentlichung	29.08.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-25019/
CVSS-Score	4.3
Finder-Beschreibung	https://security.stackexchange.com/questions/225799/dangers-of-electrons-shell-openexternal-on-untrusted-content
Zusammenfassung	Ausführung eines beliebigen Executables auf dem Server möglich, jedoch ohne Parameter usw.
Betroffene Versionen	< 2.3.0

C 2.2.2 Korrektur-Informationen

Commit **https://github.com/jitsi/jitsi-meet-electron/commit/ca1eb702507fdc4400fe21c905a9f85702f92a14**.

Der gefährliche Elektron-Aufruf `shell.openExternal` erfolgt nun erst nach der Prüfung, ob es sich beim Aufrufziel um einen HTTP(S) Link handelt (Regex `/^https?:/i`).

C 2.2.3 Bewertung

Nicht Teil von zu untersuchenden Jitsi-Bestandteilen.

C 2.3 CVE-2020-11878

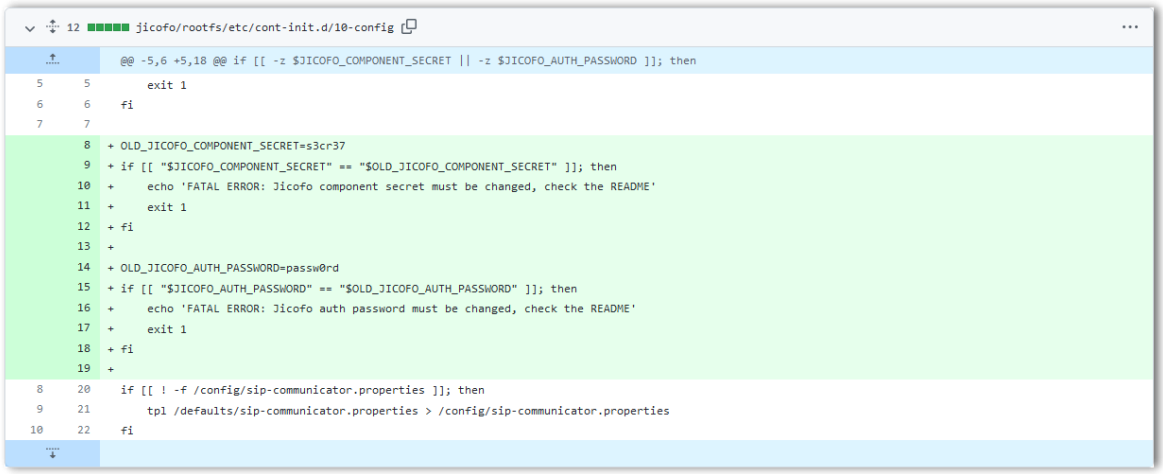
C 2.3.1 Steckbrief

Fix-Bewertung	CVE-Commit adhoc/schlecht (im aktuellen Testgegenstand jedoch solide umgesetzt)
CWE	798 : Use of Hard-coded Credentials
CVE-Kurzbeschreibung	The Jitsi Meet (aka docker-jitsi-meet) stack on Docker before stable-4384-1 uses default passwords (such as passw0rd) for system accounts.
Veröffentlichung	17.04.2020
CVE-Details	https://www.cvedetails.com/cve/CVE-2020-11878/
CVSS-Score	7.5
Finder-Beschreibung	--
Zusammenfassung	Default- bzw. Trivial-Passwörter kommen zum Einsatz
Betroffene Versionen	< stable-4384-1

C 2.3.2 Korrektur-Informationen

In Commit **https://github.com/jitsi/docker-jitsi-meet/commit/768b6c4a50d75b143ca5006e63182f1eb924076c** wird in 5 Skript-

Dateien eine Prüfung analog zur folgenden eingeführt:



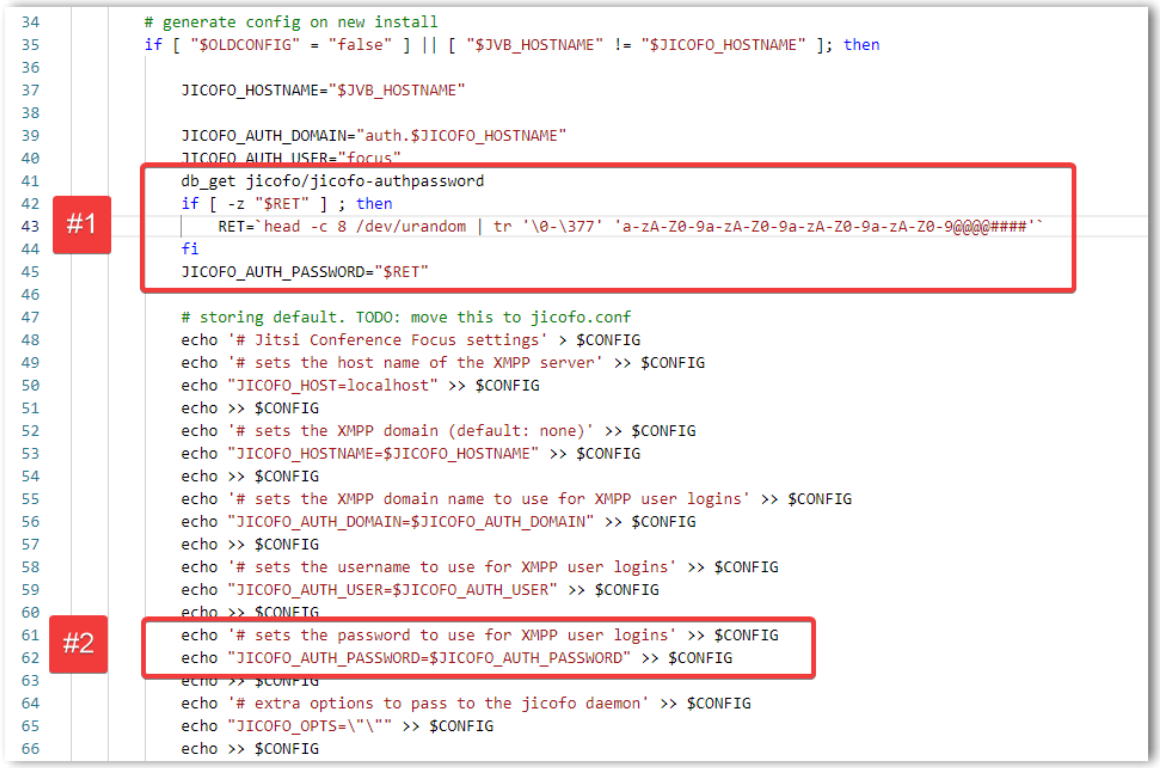
```
12 ■■■■■ jicofo/rootfs/etc/cont-init.d/10-config
@@ -5,6 +5,18 @@ if [[ -z $JICOFO_COMPONENT_SECRET || -z $JICOFO_AUTH_PASSWORD ]]; then
     exit 1
 fi

+ OLD_JICOFO_COMPONENT_SECRET=s3cr37
+ if [[ "$JICOFO_COMPONENT_SECRET" == "$OLD_JICOFO_COMPONENT_SECRET" ]]; then
+     echo 'FATAL ERROR: Jicofo component secret must be changed, check the README'
+     exit 1
+ fi
+
+ OLD_JICOFO_AUTH_PASSWORD=passw0rd
+ if [[ "$JICOFO_AUTH_PASSWORD" == "$OLD_JICOFO_AUTH_PASSWORD" ]]; then
+     echo 'FATAL ERROR: Jicofo auth password must be changed, check the README'
+     exit 1
+ fi
+
 if [[ ! -f /config/sip-communicator.properties ]]; then
     tpl /defaults/sip-communicator.properties > /config/sip-communicator.properties
 fi
```

Es wird geprüft, ob das gesetzte Passwort dem Default-Passwort entspricht. Ist dies der Fall, bricht das Skript mit Fehlercode 1 ab.

C 2.3.3 Bewertung

Diese Behandlung sieht nach einer adhoc-Lösung aus, welche vermutlich unter Zeitdruck entstanden ist und daher als suboptimal/schlecht einzustufen ist.

Hinweis: Dies hat man später wohl auch erkannt und in der untersuchten Version werden Passwörter bei Neu-Installationen zufällig generiert (bspw. `jicofo/debian/postinst`):



```
34  # generate config on new install
35  if [ "$OLDCONFIG" = "false" ] || [ "$JVB_HOSTNAME" != "$JICOFO_HOSTNAME" ]; then
36
37      JICOFO_HOSTNAME="$JVB_HOSTNAME"
38
39      JICOFO_AUTH_DOMAIN="auth.$JICOFO_HOSTNAME"
40      JICOFO_AUTH_USER="focus"
41      db_get jicofo/jicofo-authpassword
42      if [ -z "$RET" ] ; then                                          #1
43          RET=`head -c 8 /dev/urandom | tr '\0-\377' 'a-zA-Z0-9a-zA-Z0-9a-zA-Z0-9a-zA-Z0-9@@@@####'`
44      fi
45      JICOFO_AUTH_PASSWORD="$RET"
46
47      # storing default. TODO: move this to jicofo.conf
48      echo '# Jitsi Conference Focus settings' > $CONFIG
49      echo '# sets the host name of the XMPP server' >> $CONFIG
50      echo "JICOFO_HOST=localhost" >> $CONFIG
51      echo >> $CONFIG
52      echo '# sets the XMPP domain (default: none)' >> $CONFIG
53      echo "JICOFO_HOSTNAME=$JICOFO_HOSTNAME" >> $CONFIG
54      echo >> $CONFIG
55      echo '# sets the XMPP domain name to use for XMPP user logins' >> $CONFIG
56      echo "JICOFO_AUTH_DOMAIN=$JICOFO_AUTH_DOMAIN" >> $CONFIG
57      echo >> $CONFIG
58      echo '# sets the username to use for XMPP user logins' >> $CONFIG
59      echo "JICOFO_AUTH_USER=$JICOFO_AUTH_USER" >> $CONFIG
60      echo >> $CONFIG                                                  #2
61      echo '# sets the password to use for XMPP user logins' >> $CONFIG
62      echo "JICOFO_AUTH_PASSWORD=$JICOFO_AUTH_PASSWORD" >> $CONFIG
63      echo >> $CONFIG
64      echo '# extra options to pass to the jicofo daemon' >> $CONFIG
65      echo "JICOFO_OPTS=\"\"" >> $CONFIG
66      echo >> $CONFIG
```

Es wird zuerst geprüft, ob bereits ein Passwort in der BashDB extern vorgegeben wurde und falls nicht, wird über `dev/urandom` ein neues (im Beispiel oben 8-

stelliges) Passwort generiert (#1). Dieses wird dann in die Konfiguration geschrieben (#2).

D. Testergebnisse automatische Werkzeuge

Im Folgenden werden die Untersuchungsergebnisse, unterteilt nach automatischem Analysewerkzeug, zusammengefasst dargestellt. Detaillierte Ergebnisdetails können den jeweils referenzierten Ergebnisberichten entnommen werden.

D 1 Microfocus Fortify

Für diese Scans kamen der gesamte Standard-Regelsatz zum Einsatz. Folgende Versionen fanden dabei Verwendung:

- SCA Engine Version: 21.2.2.0004
- Rulepack Version: 2021.4.0.008

Die initiale Finding-Einstufung erfolgte auf Basis der Fortify Standard-Bewertung (Critical, High, Medium, Low). Die Auditierung der Findings erfolgte innerhalb der Fortify Audit-Workbench.

Im Folgenden werden die aus dem Scan gewonnenen Erkenntnisse zusammenfassend kurz dargestellt. Details können den jeweils angehängten Ergebnis-Dokumenten (PDF-Format) entnommen werden.

D 1.1 BigBlueButton

Roh-Scans, bewertete Scans und PDF-Exporte können dem Datei-Anhang entnommen werden:

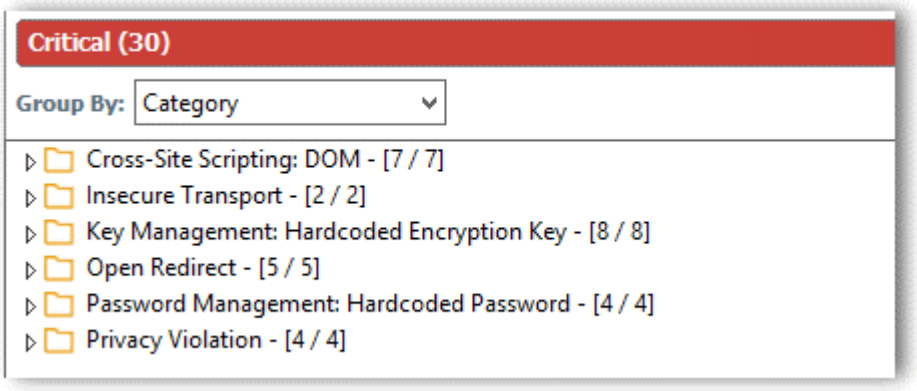
- fortify/reports/bbb/BBB_Exploitable.pdf
    - Findings, welche als „ausnutzbar“ eingestuft wurden
- fortify/reports/bbb/BBB_Suspicious.pdf
    - Findings, welche als „verdächtig“ eingestuft wurden
- fortify/reports/bbb/BBB_Bad_Practice.pdf
    - Findings, welche als „schlechte Praxis“ (meist QS-Themen) eingestuft wurden
- fortify/reports/bbb/BBB_Not_an_Issue.pdf
    - Findings, welche als „kein Problem“ eingestuft wurden
- fortify/reports/bbb/BBB_UNAUDITED.pdf

- Sämtliche Findings (nicht auditiert)

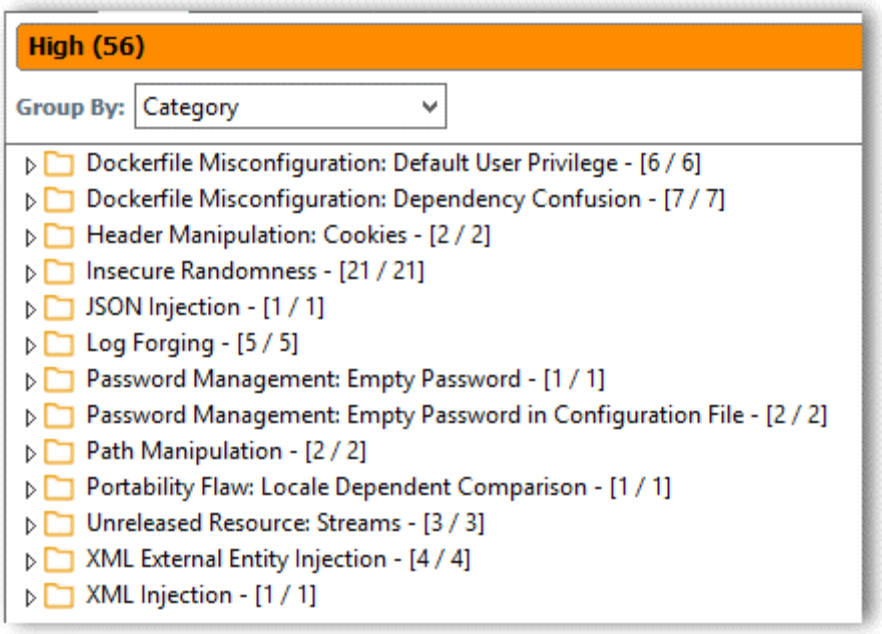
- fortify/raw/bbb/BBB_Audited.fpr
  - Toolspezifisches Format, welches die auditierten Findings enthält
- fortify/raw/bbb/BBB-UNAUDITED.fpr
  - Toolspezifisches Format, welches das initiale (unauditierte) Scan-Ergebnis enthält

D 1.1.1 Übersicht über Scan-Ergebnis

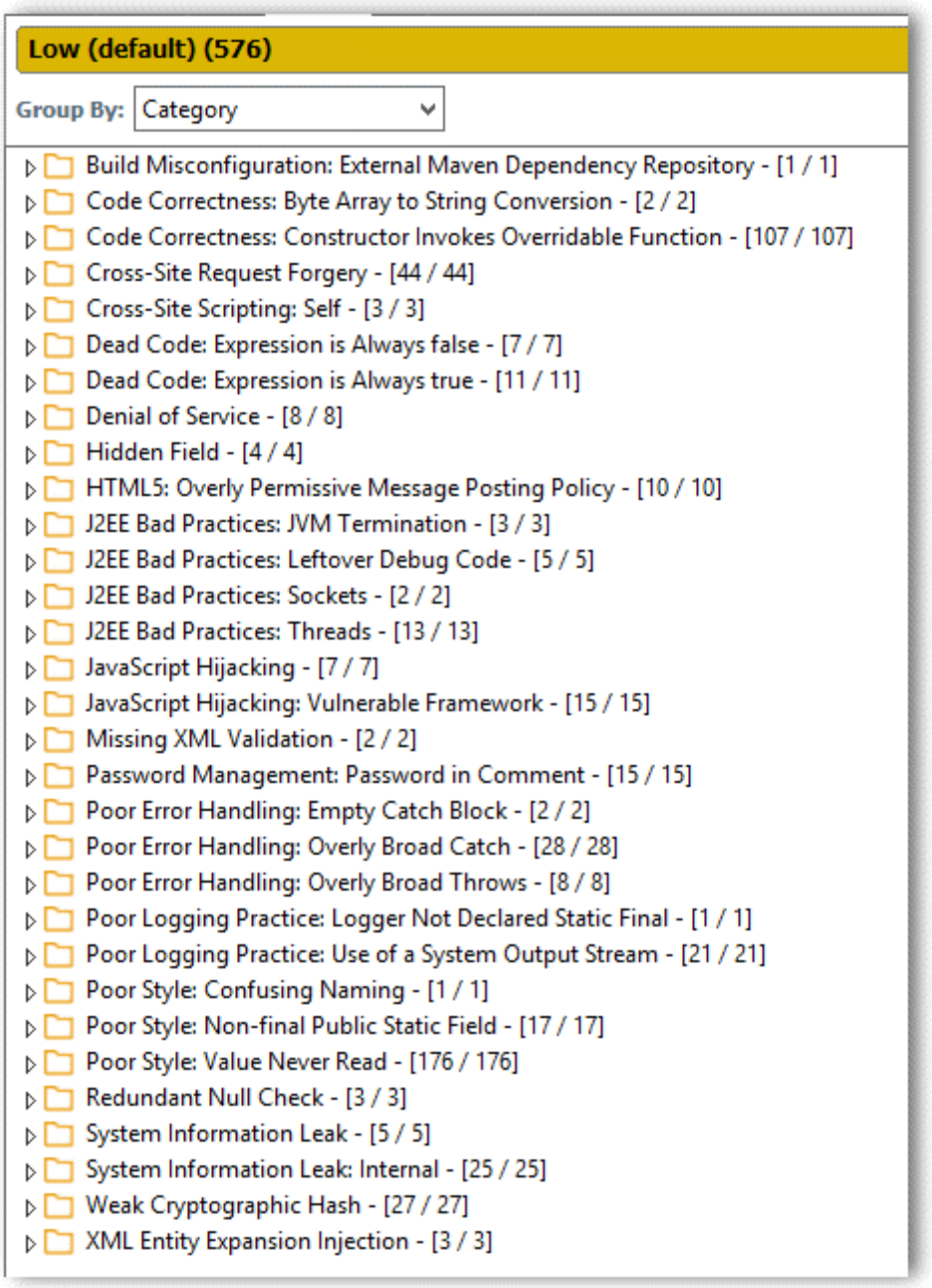
30 Findings wurden initial vom Werkzeug als „Critical“ in folgenden Kategorien und Mengen (zweite Zahlen in Klammern) zurückgeliefert:



56 als „High“ eingestuft:

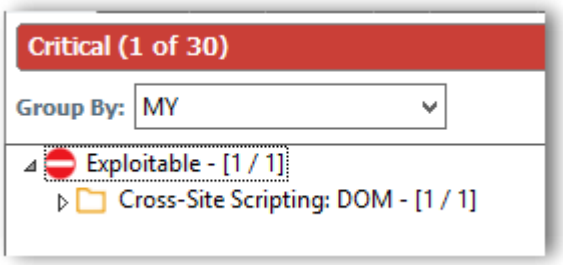


0 als „Medium“ und 576 als „Low“ eingestuft:



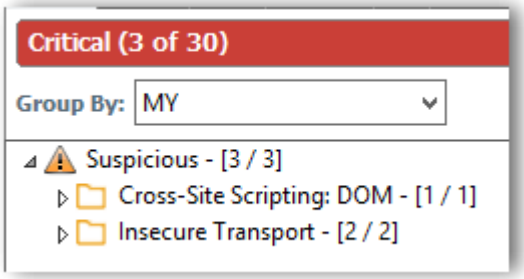
D 1.1.2 Übersicht über Bewertung durch den Auditor

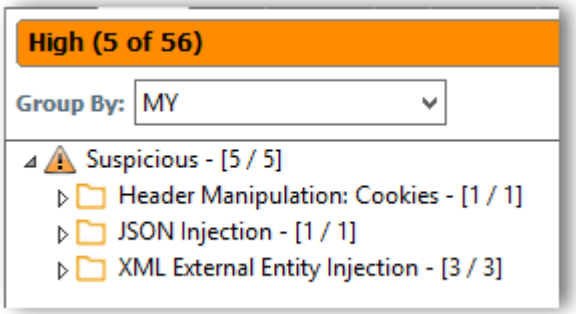
1x Einstufung als „bestätigt“

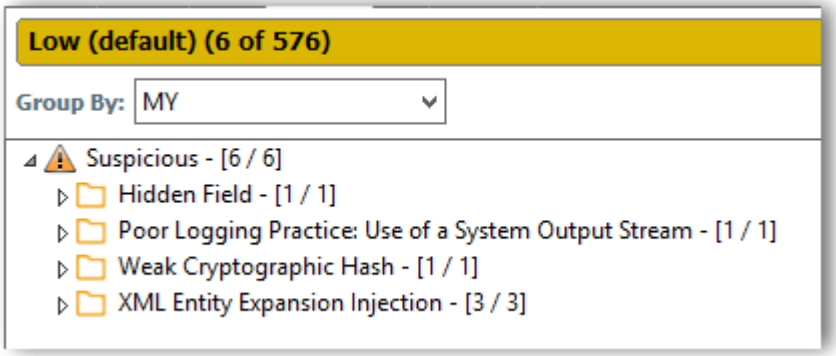


Hinweis: Das hier aufgefundene Cross-Site Scripting wurde als CVE-2022-26497 gemeldet (s. Kapitel 0).

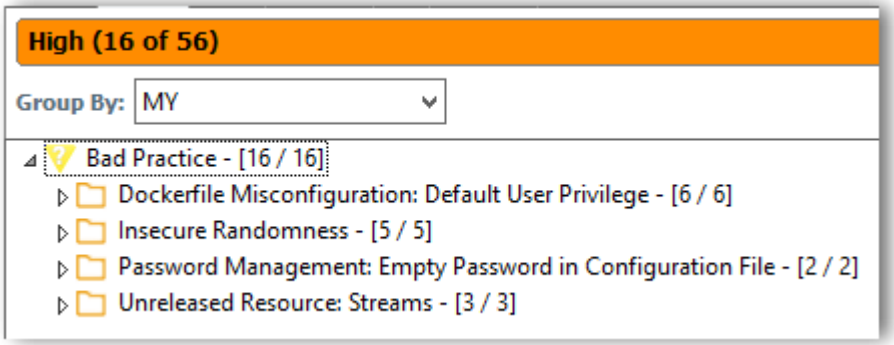
14x Einstufung als „verdächtig“

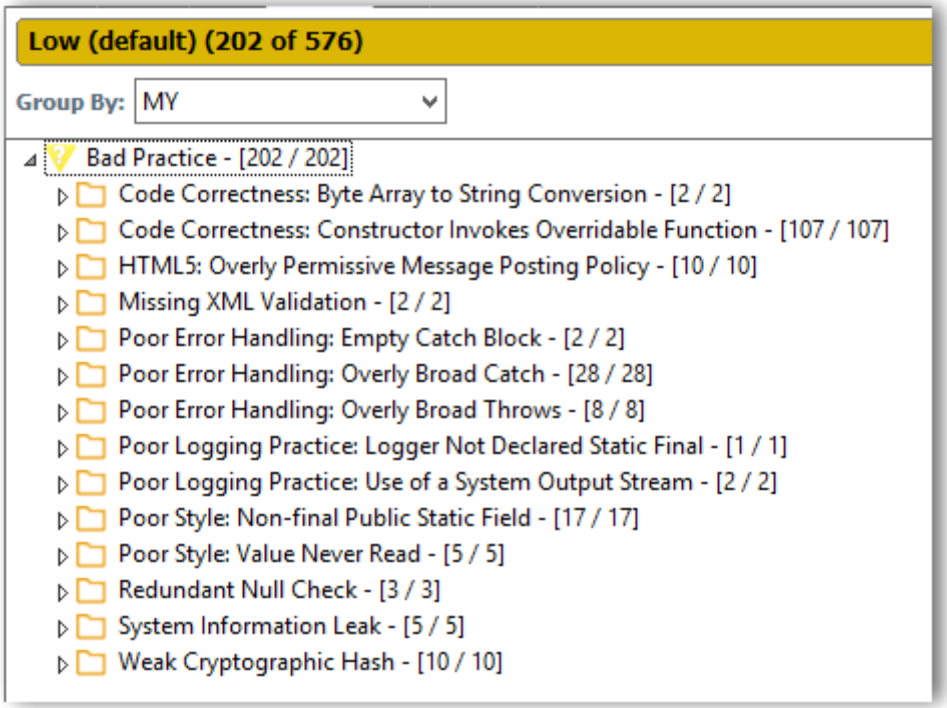




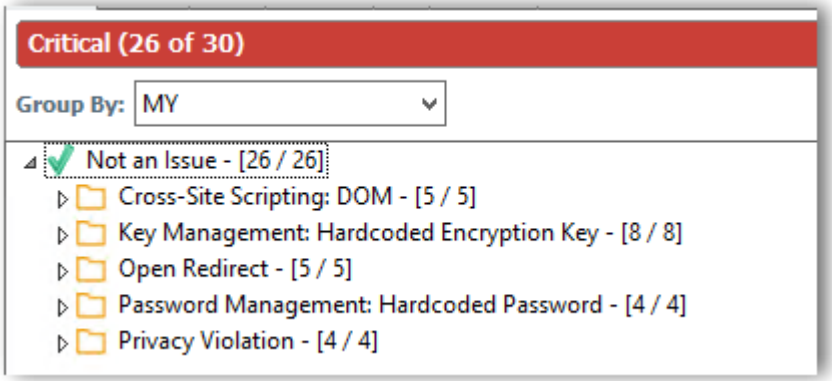


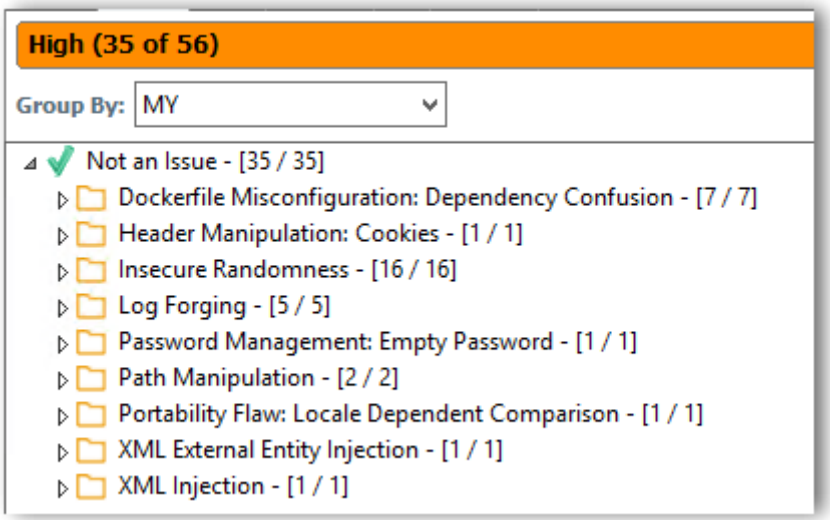
218x Einstufung als „schlechte Praxis“

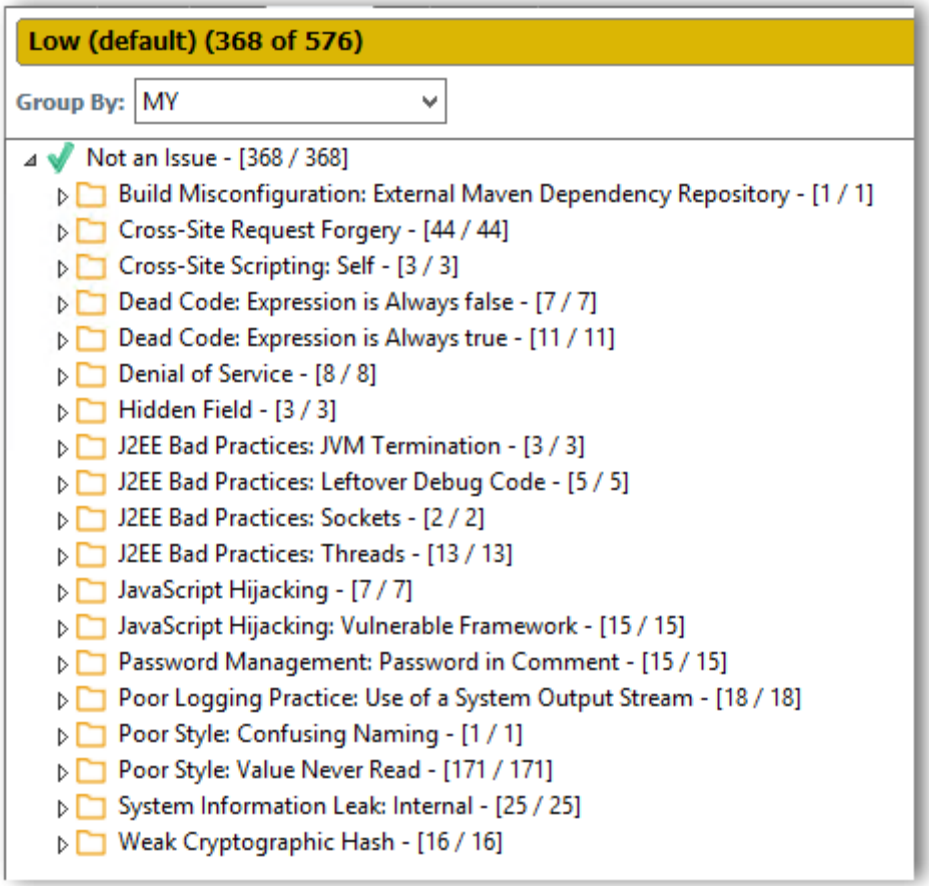




429x Einstufung als „kein Problem“







D 1.2 Jitsi

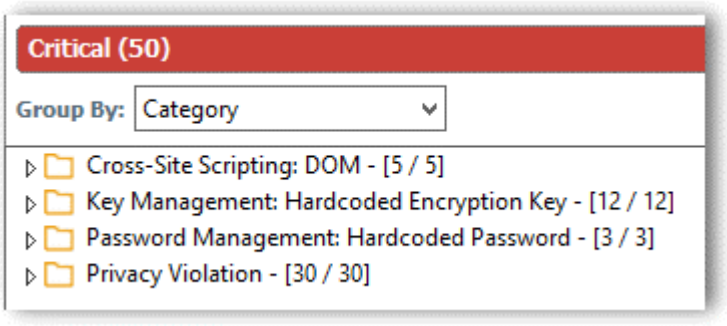
Roh-Scans, bewertete Scans und PDF-Exporte können dem Datei-Anhang entnommen werden:

- fortify/reports/jitsi/Jitsi_Exploitable.pdf
    - Findings, welche als „ausnutzbar“ eingestuft wurden
- fortify/reports/jitsi/Jitsi_Suspicious.pdf
    - Findings, welche als „verdächtig“ eingestuft wurden
- fortify/reports/jitsi /Jitsi_Bad_Practice.pdf
    - Findings, welche als „schlechte Praxis“ (meist QS-Themen) eingestuft wurden
- fortify/reports/jitsi/Jitsi_Not_an_Issue.pdf
    - Findings, welche als „kein Problem“ eingestuft wurden
- fortify/reports/jitsi/Jitsi_UNAUDITED.pdf
    - Sämtliche Findings (nicht auditiert)

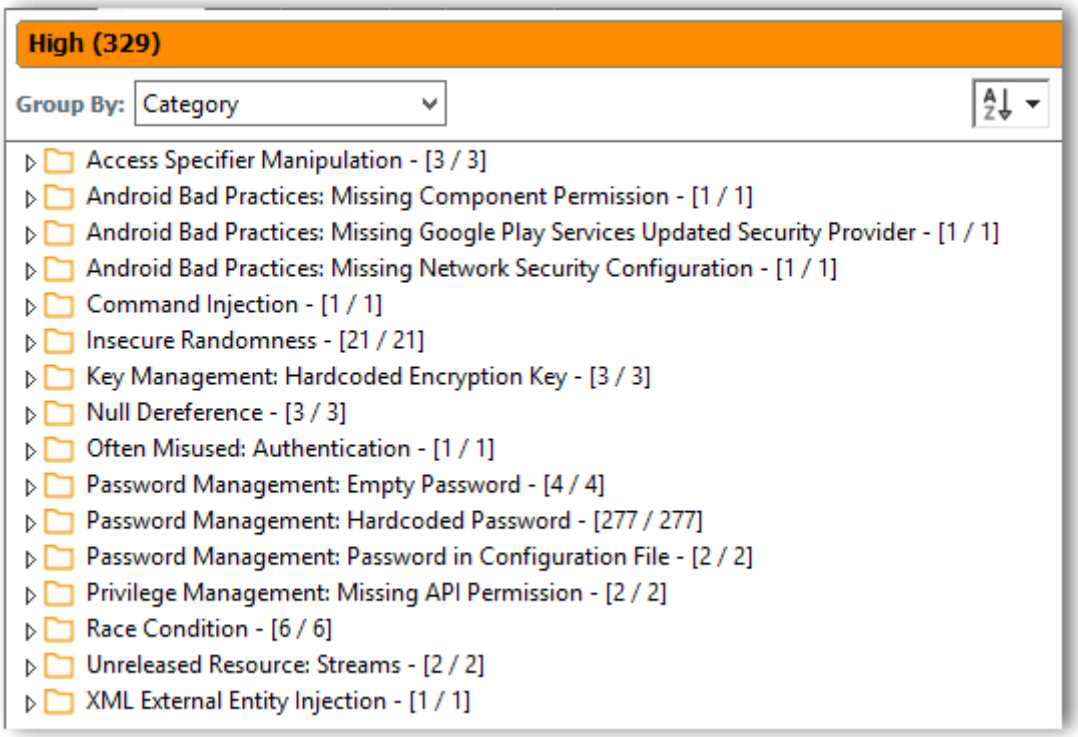
- fortify/raw/jitsi/Jitsi_Audited.fpr
    - Toolspezifisches Format, welches die auditierten Findings enthält
- fortify/raw/jitsi/Jitsi-UNAUDITED.fpr
    - Toolspezifisches Format, welches das initiale (unauditierte) Scan-Ergebnis enthält

D 1.2.1 Übersicht über Scan-Ergebnis

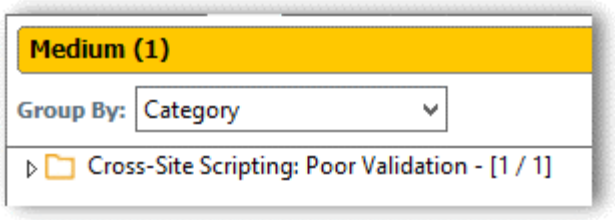
50 Findings wurden initial vom Werkzeug als „Critical“ in folgende Kategorien und Mengen (zweite Zahlen in Klammern) zurückgeliefert:



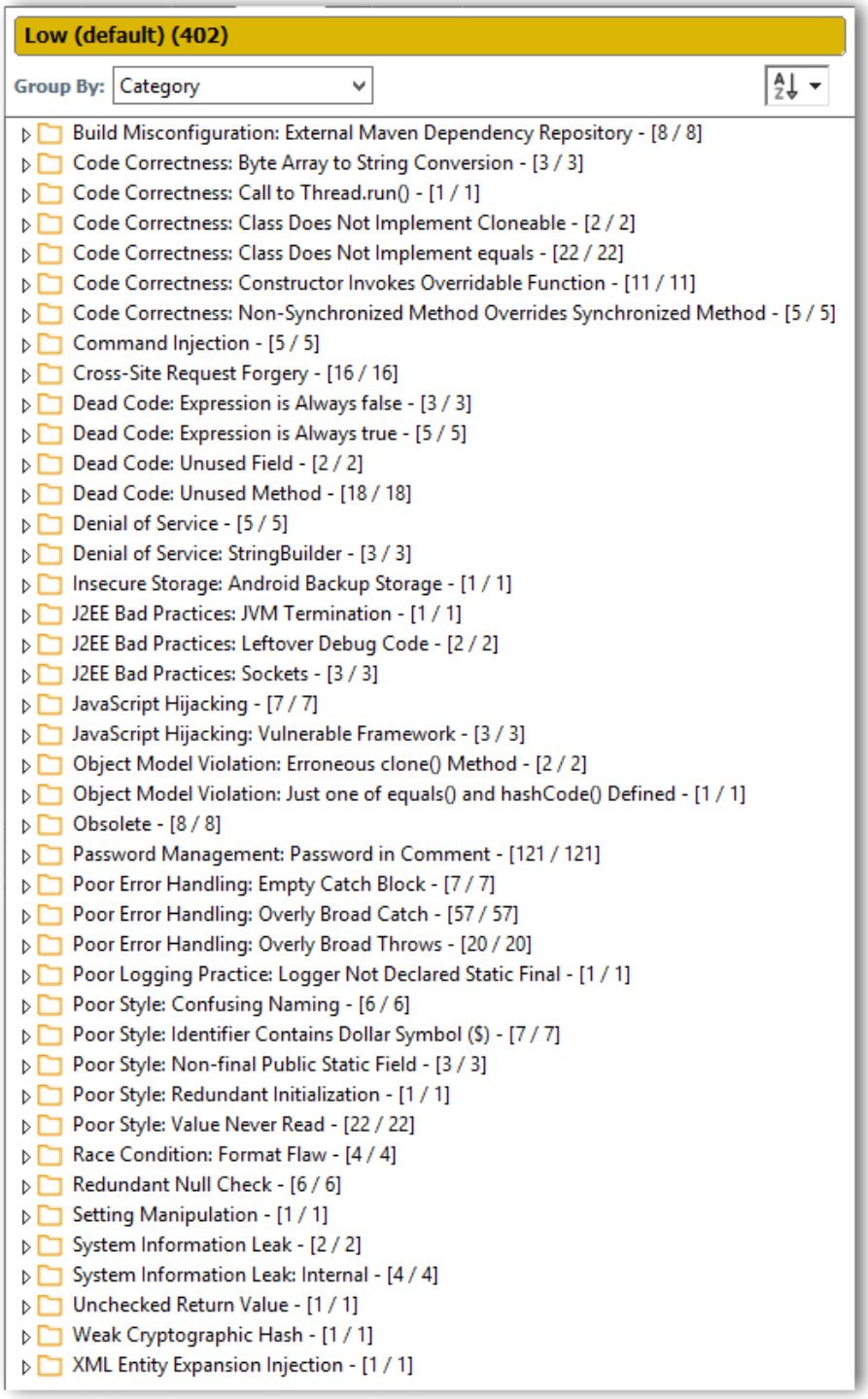
329 als „High“ eingestuft:



1 als „Medium“ eingestuft:



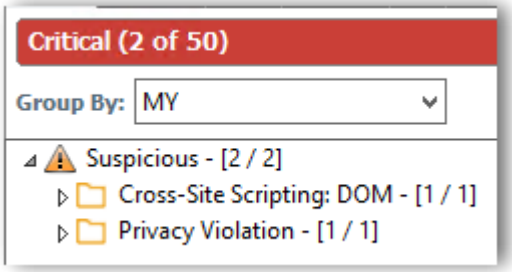
402 als „Low“ eingestuft:

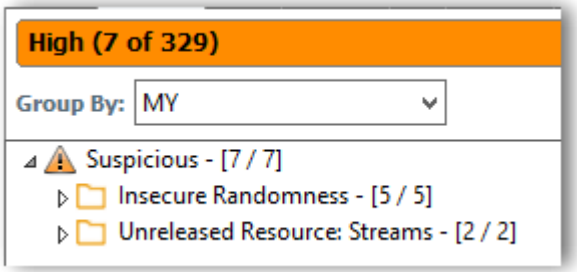


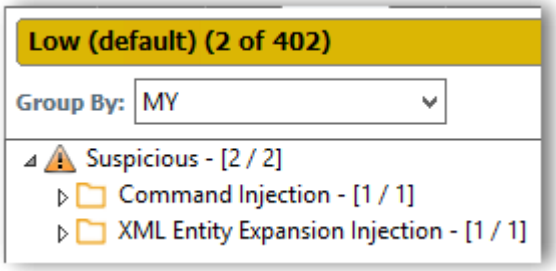
D 1.2.2 Übersicht über Bewertung durch den Auditor

0x Einstufung als „bestätigt“

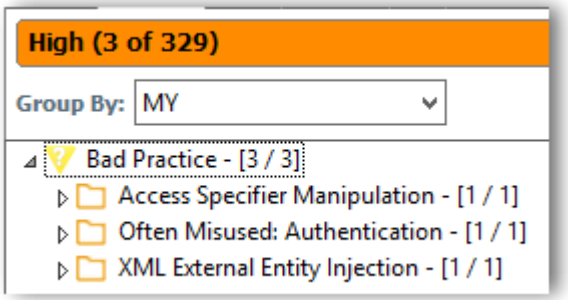
11x Einstufung als „verdächtig“

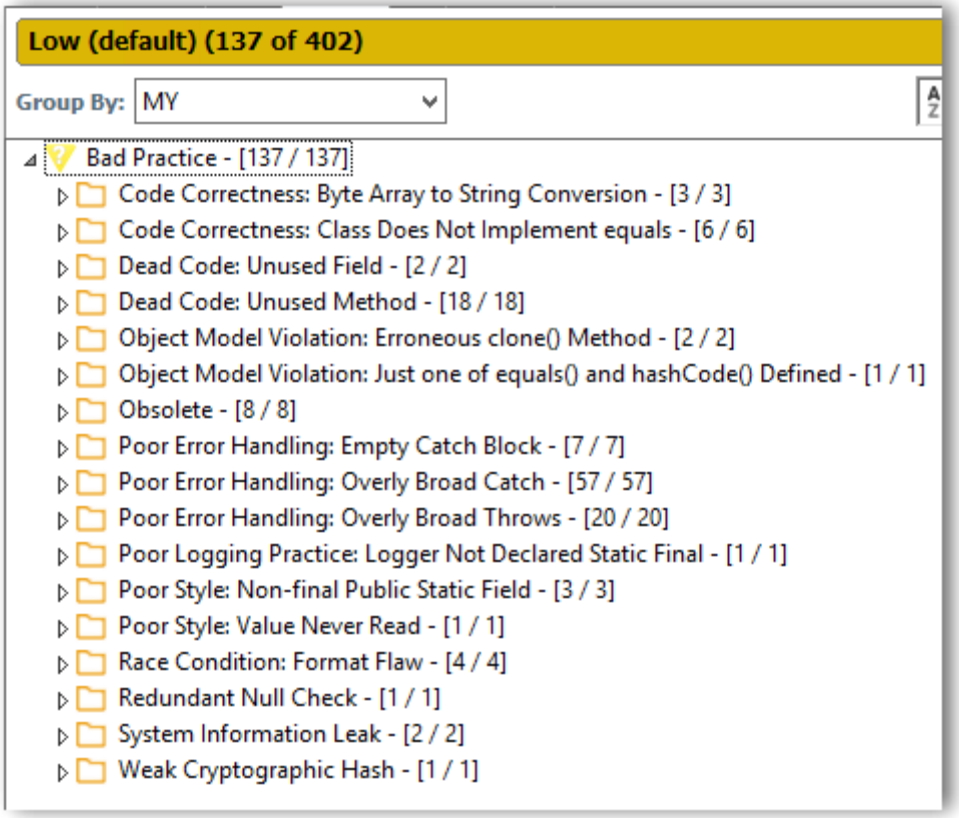




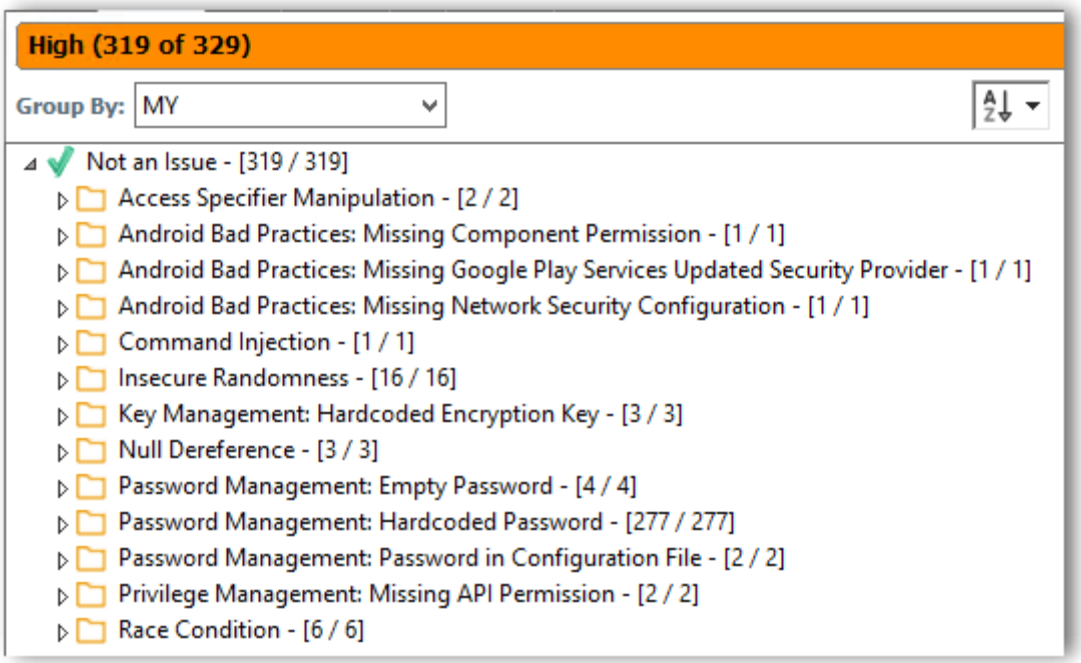


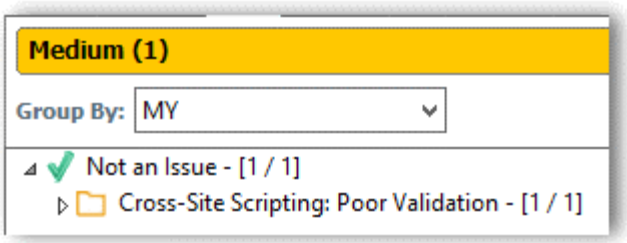
140x Einstufung als „schlechte Praxis“

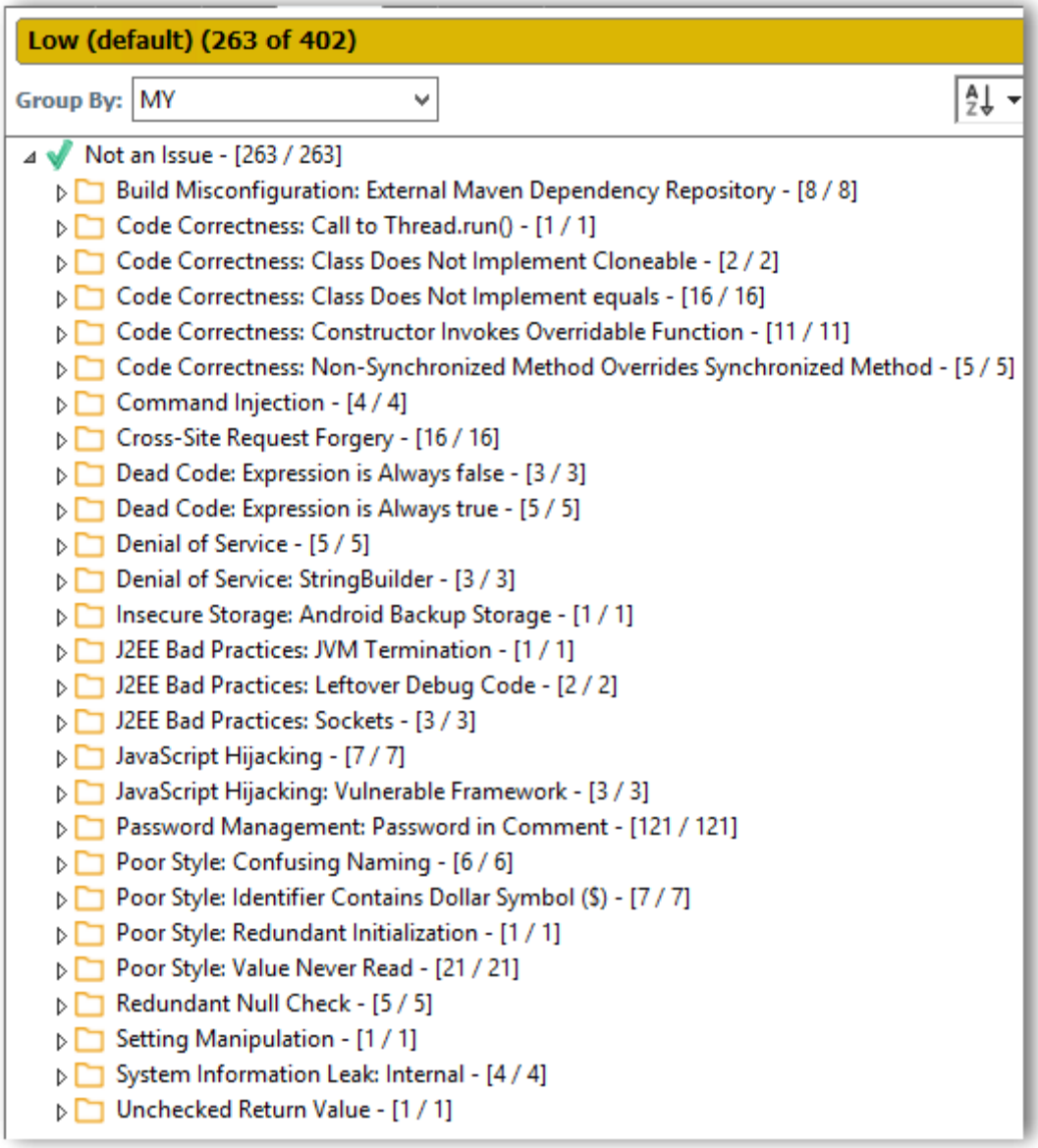




631x Einstufung als „kein Problem“


High (319 of 329)
Group By: MY
Not an Issue - [319 / 319]
Access Specifier Manipulation - [2 / 2]
Android Bad Practices: Missing Component Permission - [1 / 1]
Android Bad Practices: Missing Google Play Services Updated Security Provider - [1 / 1]
Android Bad Practices: Missing Network Security Configuration - [1 / 1]
Command Injection - [1 / 1]
Insecure Randomness - [16 / 16]
Key Management: Hardcoded Encryption Key - [3 / 3]
Null Dereference - [3 / 3]
Password Management: Empty Password - [4 / 4]
Password Management: Hardcoded Password - [277 / 277]
Password Management: Password in Configuration File - [2 / 2]
Privilege Management: Missing API Permission - [2 / 2]
Race Condition - [6 / 6]


Medium (1)
Group By: MY
Not an Issue - [1 / 1]
Cross-Site Scripting: Poor Validation - [1 / 1]


Low (default) (263 of 402)
Group By: MY
Not an Issue - [263 / 263]
Build Misconfiguration: External Maven Dependency Repository - [8 / 8]
Code Correctness: Call to Thread.run() - [1 / 1]
Code Correctness: Class Does Not Implement Cloneable - [2 / 2]
Code Correctness: Class Does Not Implement equals - [16 / 16]
Code Correctness: Constructor Invokes Overridable Function - [11 / 11]
Code Correctness: Non-Synchronized Method Overrides Synchronized Method - [5 / 5]
Command Injection - [4 / 4]
Cross-Site Request Forgery - [16 / 16]
Dead Code: Expression is Always false - [3 / 3]
Dead Code: Expression is Always true - [5 / 5]
Denial of Service - [5 / 5]
Denial of Service: StringBuilder - [3 / 3]
Insecure Storage: Android Backup Storage - [1 / 1]
J2EE Bad Practices: JVM Termination - [1 / 1]
J2EE Bad Practices: Leftover Debug Code - [2 / 2]
J2EE Bad Practices: Sockets - [3 / 3]
JavaScript Hijacking - [7 / 7]
JavaScript Hijacking: Vulnerable Framework - [3 / 3]
Password Management: Password in Comment - [121 / 121]
Poor Style: Confusing Naming - [6 / 6]
Poor Style: Identifier Contains Dollar Symbol ($) - [7 / 7]
Poor Style: Redundant Initialization - [1 / 1]
Poor Style: Value Never Read - [21 / 21]
Redundant Null Check - [5 / 5]
Setting Manipulation - [1 / 1]
System Information Leak: Internal - [4 / 4]
Unchecked Return Value - [1 / 1]

D 2 Checkmarx SAST

Für diese Scans kamen die On-Demand-Plattform von Checkmarx (AST) mit deren Standard-Regelsatz zum Einsatz. Die genauen Engine- und Rulepack-Versionen konnten nicht herausgefunden werden, entsprachen aber, nach Angabe des Herstellers, den zum Zeitpunkt der Analyse (22.03.2022) aktuellsten Produktiv-Versionen.

Die initiale Finding-Einstufung erfolgte auf Basis der Checkmarx Standard-Bewertung (High, Medium, Low, Info). Die Auditierung der Findings erfolgte innerhalb der On-Demand Plattform (AST).

Im Folgenden werden die aus dem Scan gewonnenen Erkenntnisse zusammenfassend kurz dargestellt. Details können den jeweils angehängten Ergebnis-Dokumenten (PDF, HTML, JSON- und SARIF-Format) entnommen werden.

Hinweis: Die verschiedenen RAW Tool-Exporte unterscheiden sich in der Menge der Findings zu den jeweils aufgezeigten Übersichts- und Bewertungsmengen in den folgenden Abschnitten. Dies liegt daran, dass von Checkmarx nur die SAST-Ergebnisse in diesen Bericht übernommen wurden. Die SCA-Ergebnisse unterscheiden sich nicht erwähnenswert von denen des frei verfügbaren DependencyCheck-Ergebnisses (s. Kapitel D5). Die KICS-Ergebnisse liegen ausserhalb des Scopes dieser Analyse.

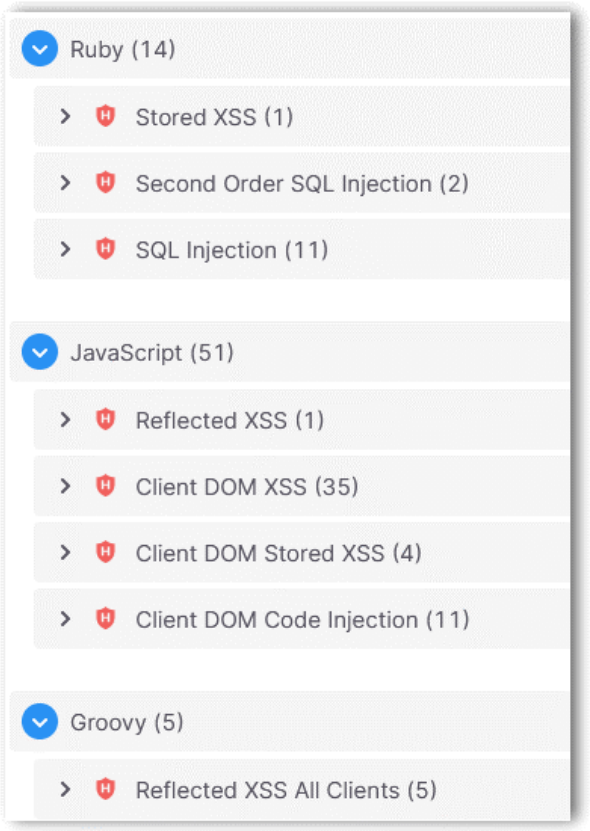
D 2.1 BigBlueButton

Roh-Scans und bewertete Scans können dem Datei-Anhang entnommen werden:

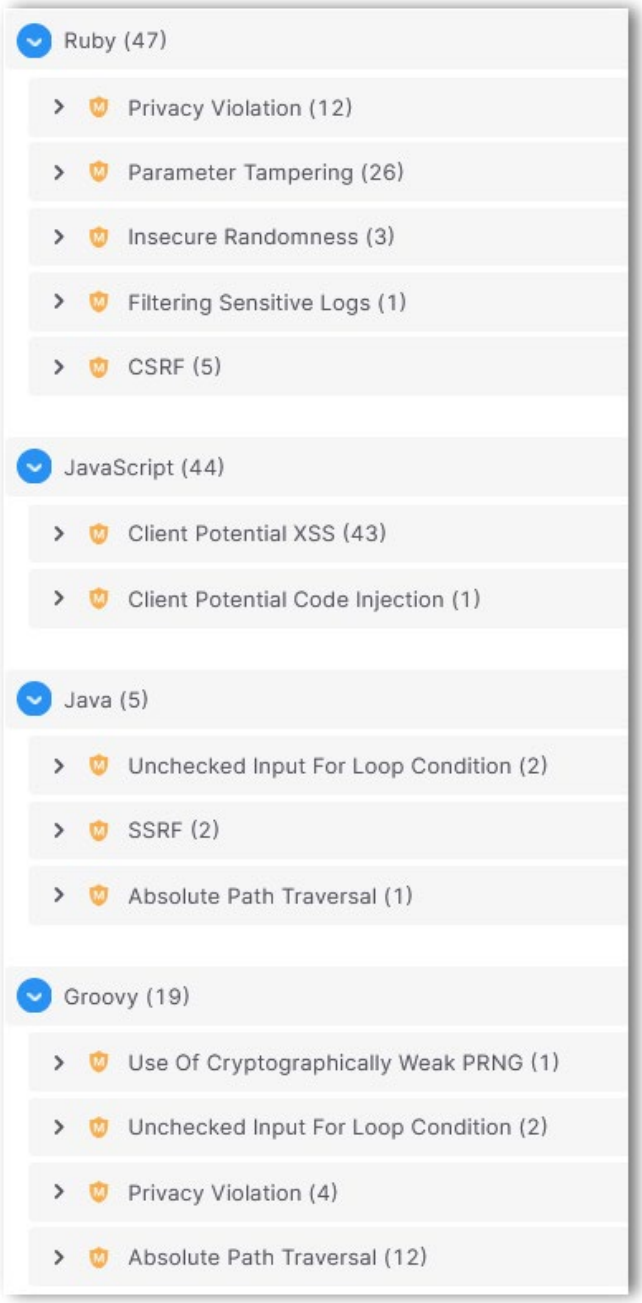
- checkmarx/reports/bbb/BBB-SUMMARY.html
    - Kurzübersicht über Finding-Mengen
- checkmarx/reports/bbb/BBB-FULL.pdf
    - Alle Findings inkl. Bewertung
- checkmarx/raw/bbb/bbb.json
    - Alle Findings inkl. Bewertung im JSON-Format
- checkmarx/raw/bbb/bbb.sarif
    - Alle Findings inkl. Bewertung im SARIF-Format
- checkmarx/raw/bbb/by_state/bbb-CONFIRMED.[json|sarif]
    - Findings, welche als „bestätigt/ausnutzbar“ eingestuft wurden (im JSON- bzw. SARIF-Format)
- checkmarx/raw/bbb/by_state/bbb-NOT_EXPLOITABLE.[json|sarif]
    - Findings, welche als „nicht ausnutzbar/kein Problem“ eingestuft wurden (im JSON- bzw. SARIF-Format)

D 2.1.1 Übersicht über Scan-Ergebnis

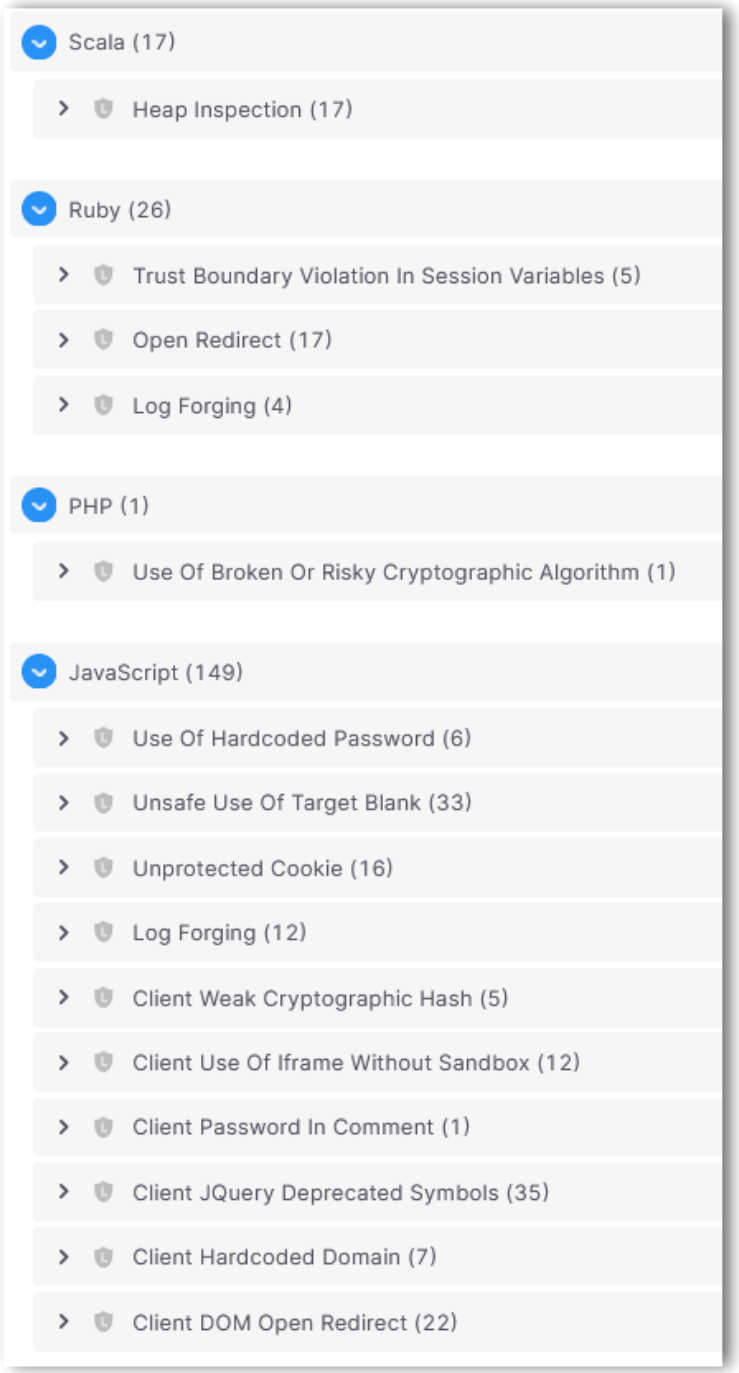
70 Findings wurden initial vom Werkzeug als „High“ in folgenden Kategorien und Mengen (Zahl in Klammern) zurückgeliefert:

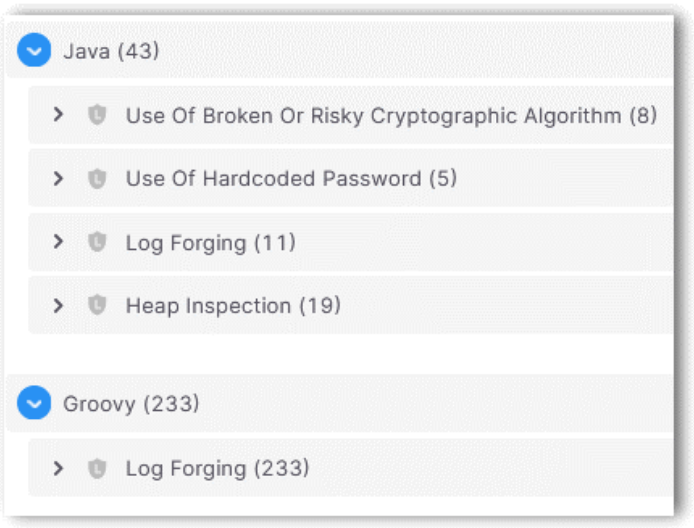


115 als Medium:


Ruby (47)
Privacy Violation (12)
Parameter Tampering (26)
Insecure Randomness (3)
Filtering Sensitive Logs (1)
CSRF (5)
JavaScript (44)
Client Potential XSS (43)
Client Potential Code Injection (1)
Java (5)
Unchecked Input For Loop Condition (2)
SSRF (2)
Absolute Path Traversal (1)
Groovy (19)
Use Of Cryptographically Weak PRNG (1)
Unchecked Input For Loop Condition (2)
Privacy Violation (4)
Absolute Path Traversal (12)

469 als Low:

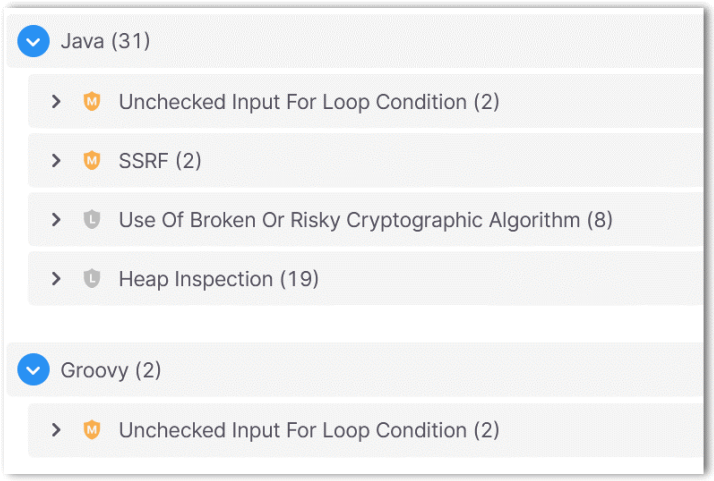

Scala (17)
Heap Inspection (17)
Ruby (26)
Trust Boundary Violation In Session Variables (5)
Open Redirect (17)
Log Forging (4)
PHP (1)
Use Of Broken Or Risky Cryptographic Algorithm (1)
JavaScript (149)
Use Of Hardcoded Password (6)
Unsafe Use Of Target Blank (33)
Unprotected Cookie (16)
Log Forging (12)
Client Weak Cryptographic Hash (5)
Client Use Of Iframe Without Sandbox (12)
Client Password In Comment (1)
Client JQuery Deprecated Symbols (35)
Client Hardcoded Domain (7)
Client DOM Open Redirect (22)



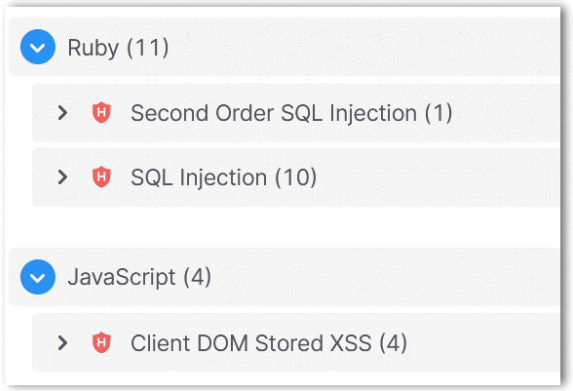
0 als Info.

D 2.1.2 Übersicht über Bewertung durch den Auditor

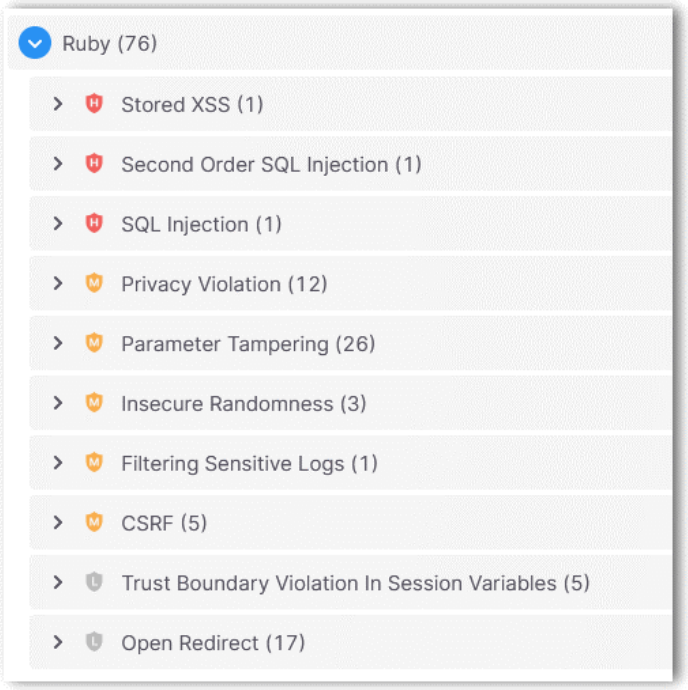
70x Einstufung als „bestätigt“

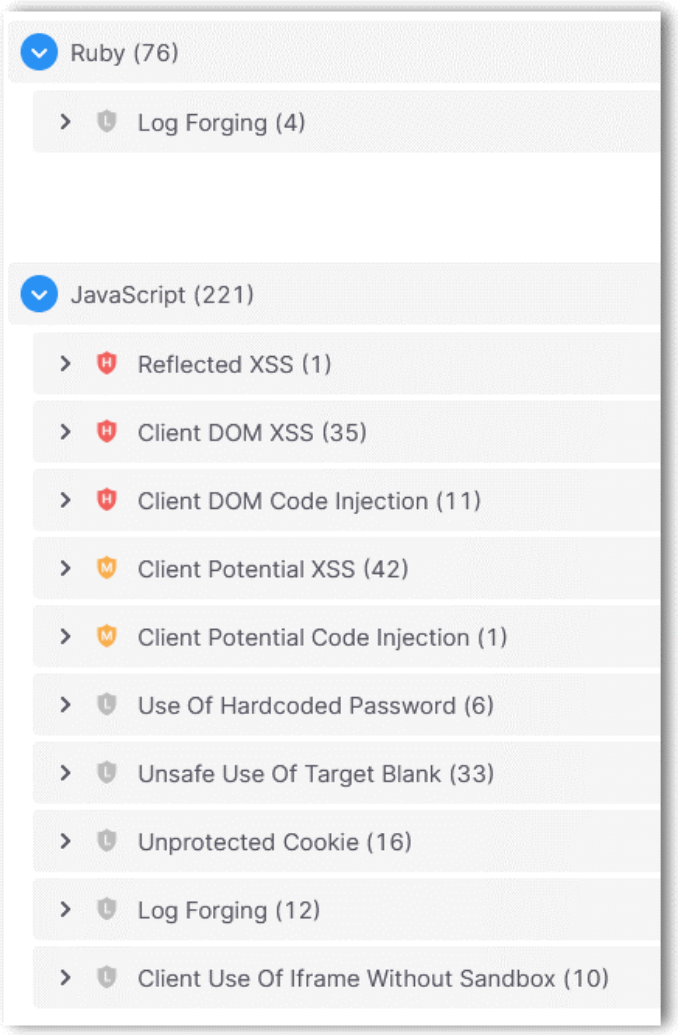


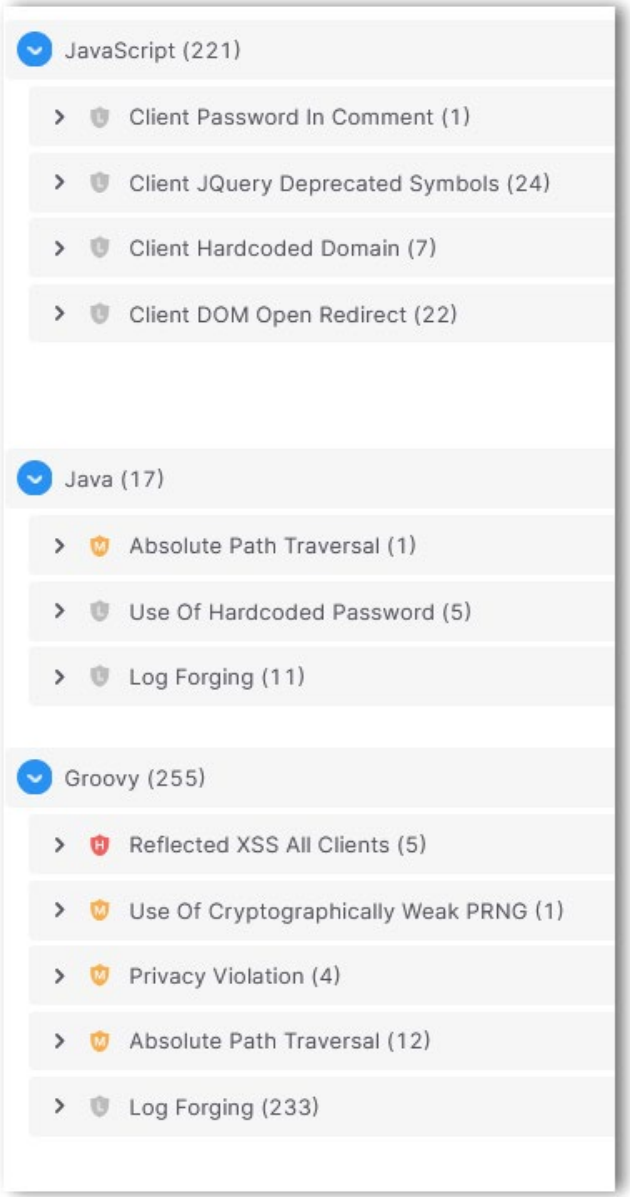
15x Einstufung als „verdächtig“



569x Einstufung als „kein Problem“




Ruby (76)
Log Forging (4)
JavaScript (221)
Reflected XSS (1)
Client DOM XSS (35)
Client DOM Code Injection (11)
Client Potential XSS (42)
Client Potential Code Injection (1)
Use Of Hardcoded Password (6)
Unsafe Use Of Target Blank (33)
Unprotected Cookie (16)
Log Forging (12)
Client Use Of Iframe Without Sandbox (10)



D 2.2 Jitsi

Roh-Scans und bewertete Scans können dem Datei-Anhang entnommen werden:

- checkmarx/reports/jitsi/Jitsi-SUMMARY.html
  - Kurzübersicht über Finding-Mengen
- checkmarx/reports/jitsi/Jitsi-FULL.pdf
  - Alle Findings inkl. Bewertung
- checkmarx/raw/jitsi/jitsi.json
  - Alle Findings inkl. Bewertung im JSON-Format

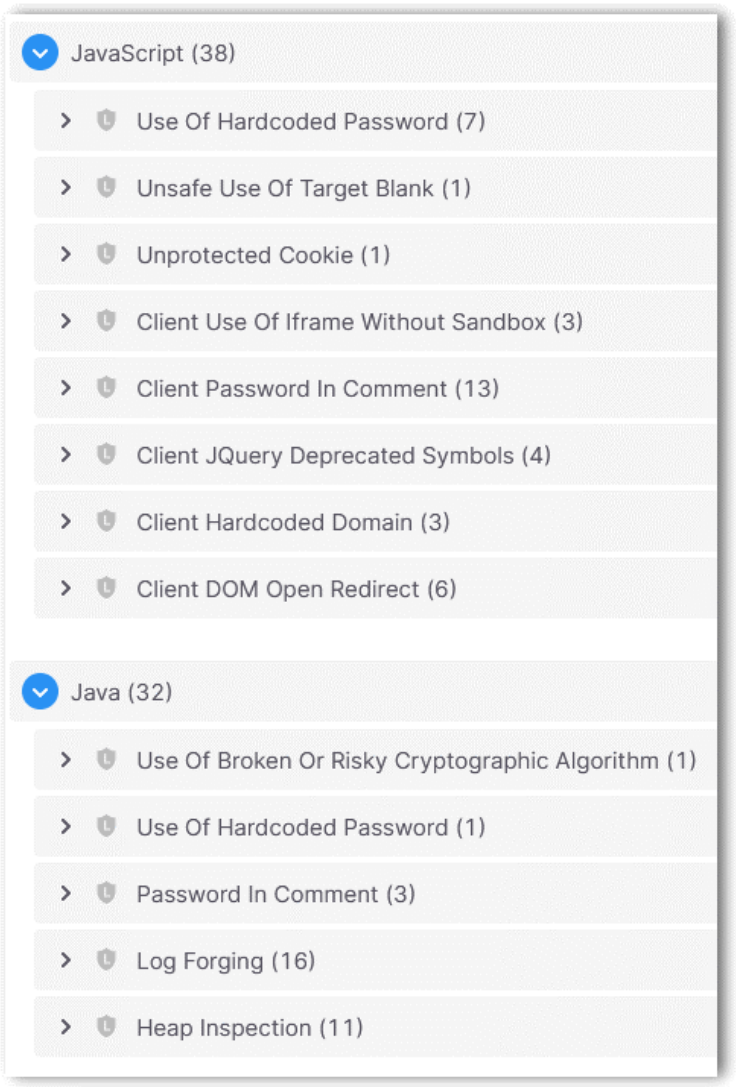
- checkmarx/raw/jitsi/jitsi.sarif
    - Alle Findings inkl. Bewertung im SARIF-Format
- checkmarx/raw/jitsi/by_state/jitsi-CONFIRMED.[json|sarif]
    - Findings, welche als „bestätigt/ausnutzbar“ eingestuft wurden (im JSON- bzw. SARIF-Format)
- checkmarx/raw/jitsi/by_state/jitsi-NOT_EXPLOITABLE.[json|sarif]
    - Findings, welche als „nicht ausnutzbar/kein Problem“ eingestuft wurden (im JSON- bzw. SARIF-Format)

D 2.2.1 Übersicht über Scan-Ergebnis

5 Findings wurden initial vom Werkzeug als „High“ in folgenden Kategorien und Mengen (Zahl in Klammern) zurückgeliefert:

9 als Medium:

70 als Low:

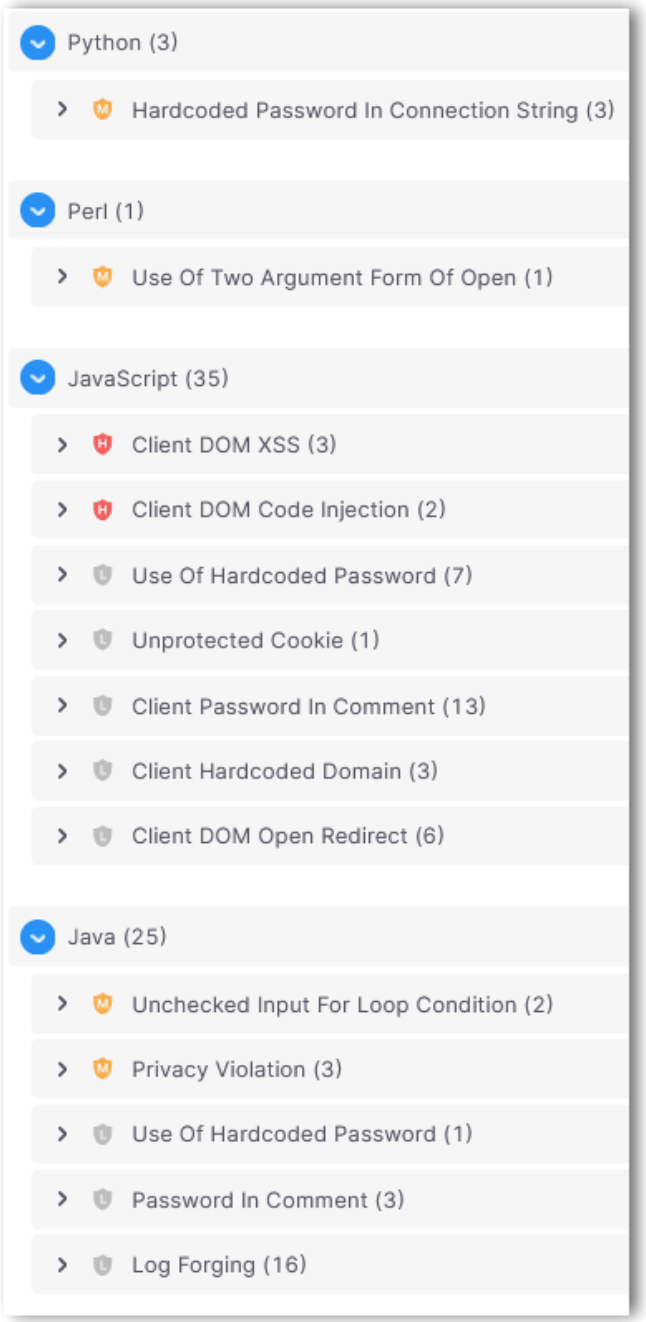


0 als Info

D 2.2.2 Übersicht über Bewertung durch den Auditor

20x Einstufung als „bestätigt“

64x Einstufung als „kein Problem“



D 3 CodeQL

Für diese Scans kamen der zur jeweiligen Sprache passende Standard-Regelsatz zum Einsatz, welcher mit dem Werkzeug ausgeliefert wurde. Folgende Versionen fanden dabei Verwendung:

- CodeQL Version: 2.8.1
- Rulepack Version: 2.8.1

Die initiale Finding-Einstufung erfolgte auf Basis der CodeQL Standard-Bewertung (error, warning, recommendation). Die Sarif-Ergebnis-Dateien wurde für die Auditierung in ein einzelnes Excel-Dokument überführt, in welchem die Bewertung erfolgte. Dieses Dokument erfüllt gleichzeitig die Aufgabe des Ergebnis-Reports.

Im Folgenden werden die aus dem Scan gewonnenen Erkenntnisse zusammenfassend kurz dargestellt. Details können den jeweils angehängten Ergebnis-Dokumenten (Excel-Format) entnommen werden.

Hinweis: Bei der Auswertung der CodeQL-Findings sind uns Code-Stellen aufgefallen, welche wie farlässige Programmierfehler aussehen und möglicherweise einen Security-Impact haben könnten. Da wir dies schlussendlich nicht zweifelsfrei bewerten konnten, haben wir hier eine Bewertungskategory „Bug“ eingeführt.

D 3.1 BigBlueButton

Die Roh-Scans, sowie das bewertete Excel-Dokument können dem Datei-Anhang entnommen werden:

- codeql/reports/bbb/bbb_AUDITED.xlsx
    - Komplettauflistung aller Findings inkl. Bewertung (enthält die Ergebnisse aller folgenden SARIF-Dateien, jedoch mit für den Audit ausreichender, etwas reduzierter Detaillierung)
- codeql/raw/bbb/bbb-events.sarif
    - unauditiertes SARIF-Format aller Findings, welche im Modul „events“ (unter Anwendung der Ruby-Regeln) aufgefunden wurden
      Hinweis: KEINE Findings
- codeql/raw/bbb/bbb-pads.sarif
    - unauditiertes SARIF-Format aller Findings, welche im Modul „pads“ (unter Anwendung der JS-Regeln) aufgefunden wurden
      Hinweis: KEINE Findings
- codeql/raw/bbb/bbb-webhooks.sarif
    - unauditiertes SARIF-Format aller Findings, welche im Modul „webhooks“ (unter Anwendung der JS-Regeln) aufgefunden wurden
- codeql/raw/bbb/bbb-webrtc-sfu.sarif

  - unauditiertes SARIF-Format aller Findings, welche im Modul „webrtc-sfu“ (unter Anwendung der JS-Regeln) aufgefunden wurden

- codeql/raw/bbb/bigbluebutton-html5.sarif
  - unauditiertes SARIF-Format aller Findings, welche im Modul „html5“ (unter Anwendung der JS-Regeln) aufgefunden wurden
- codeql/raw/bbb/greenlight.sarif
  - unauditiertes SARIF-Format aller Findings, welche im Modul „greenlight“ (unter Anwendung der Ruby-Regeln) aufgefunden wurden
- codeql/raw/bbb/greenlight-javascript.sarif
  - unauditiertes SARIF-Format aller Findings, welche im Modul „greenlight“ (unter Anwendung der JS-Regeln) aufgefunden wurden
- codeql/raw/bbb/record-and-playback-core.sarif
  - unauditiertes SARIF-Format aller Findings, welche im Modul „record-and-playback-core“ (unter Anwendung der Ruby-Regeln) aufgefunden wurden

D 3.1.1 Übersicht über Scan-Ergebnis

14 Findings wurden initial vom Werkzeug als „Error“ in folgenden Kategorien und Mengen zurückgeliefert:

- 2x rb/hardcoded-credentials
- 1x js/request-forgery
- 8x js/log-injection
- 1x js/redos
- 2x js/mixed-static-instance-this-access

114 mit “Warning”:

- 3x rb/csrf-protection-disabled
- 5x js/missing-variable-declaration
- 1x js/polynomial-redos
- 22x js/useless-assignment-to-local
- 8x js/comparison-between-incompatible-types
- 5x js/superfluous-trailing-arguments

- 4x js/react/direct-state-mutation
- 6x js/react/inconsistent-state-update
- 2x js/react/unused-or-undefined-state-property
- 1x js/regex/duplicate-in-character-class
- 3x js/insecure-randomness
- 1x js/unreachable-statement
- 12x js/trivial-conditional
- 1x js/regex/missing-regexp-anchor
- 2x js/file-system-race
- 38x js/useless-expression

141 mit “Recommendation”

- 49x js/unused-local-variable
- 5x js/unneeded-defensive-code
- 85x js/automatic-semicolon-insertion
- 2x js/syntax-error

D 3.1.2 Übersicht über Bewertung durch den Auditor

212x Einstufung als „bestätigt“

- 85x js/automatic-semicolon-insertion
- 5x js/missing-variable-declaration
- 4x js/react/direct-state-mutation
- 2x js/react/unused-or-undefined-state-property
- 1x js/regex/duplicate-in-character-class
- 1x js/regex/missing-regexp-anchor
- 5x js/superfluous-trailing-arguments
- 49x js/unused-local-variable
- 22x js/useless-assignment-to-local
- 38x js/useless-expression

2x Einstufung als „verdächtig“

- 2x js/mixed-static-instance-this-access

8x Einstufung als “schlechte Praxis ”

- 8x js/comparison-between-incompatible-types

47x Einstufung als „kein Problem“

- 2x js/file-system-race
- 3x js/insecure-randomness
- 8x js/log-injection
- 1x js/polynomial-redos
- 6x js/react/inconsistent-state-update
- 1x js/redos
- 1x js/request-forgery
- 2x js/syntax-error
- 12x js/trivial-conditional
- 5x js/unneeded-defensive-code
- 1x js/unreachable-statement
- 3x rb/csrf-protection-disabled
- 2x rb/hardcoded-credentials

D 3.2 Jitsi

Die Roh-Scans, sowie das bewertete Excel-Dokument können dem Datei-Anhang entnommen werden:

- codeql/reports/jitsi/jitsi_AUDITED.xlsx
    - Komplettauflistung aller Findings inkl. Bewertung (enthält die Ergebnisse aller folgendem SARIF-Dateien, jedoch mit für den Audit ausreichender, etwas reduzierter Detaillierung)
- codeql/raw/jitsi/jitsi-jicofo-java.sarif
    - unauditiertes SARIF-Format aller Findings, welche im Modul „jicofo“ (unter Anwendung der Java-Regeln) aufgefunden wurden
- codeql/raw/jitsi/jitsi-meet-JS.sarif

    - unauditiertes SARIF-Format aller Findings, welche im Modul „jitsi-meet“ (unter Anwendung der JS-Regeln) aufgefunden wurden

- codeql/raw/jitsi/jitsi-videobridge2-java.sarif
    - unauditiertes SARIF-Format aller Findings, welche im Modul „videobridge2“ (unter Anwendung der Java-Regeln) aufgefunden wurden

- codeql/raw/jitsi/jitsi-jibri-java.sarif
    - unauditiertes SARIF-Format aller Findings, welche im Modul „jibri“ (unter Anwendung der Java-Regeln) aufgefunden wurden

- codeql/raw/jitsi/jitsi-jigasi-java.sarif
    - unauditiertes SARIF-Format aller Findings, welche im Modul „jigasi“ (unter Anwendung der Java-Regeln) aufgefunden wurden

D 3.2.1 Übersicht über Scan-Ergebnis

11 Findings wurden initial vom Werkzeug als „Error“ in folgenden Kategorien und Mengen zurückgeliefert:

- 2x java/equals-on-unrelated-types
- 1x java/hashing-without-hashcode
- 1x java/inconsistent-equals-and-hashcode
- 1x java/type-mismatch-access
- 1x java/unsynchronized-getter
- 1x java/unused-container
- 1x java/xss
- 1x js/log-injection
- 1x js/unbound-event-handler-receiver
- 1x js/xss

77 mit “Warning”:

- 10x java/dereferenced-value-may-be-null
- 1x java/inconsistent-compareto-and-equals
- 2x java/non-null-boxed-variable
- 1x java/notify-instead-of-notify-all
- 5x java/random-used-once

- 1x java/relative-path-command
- 1x java/type-bound-extends-final
- 2x java/unused-format-argument
- 5x java/useless-null-check
- 1x java/weak-cryptographic-algorithm
- 2x js/comparison-between-incompatible-types
- 1x js/incomplete-url-substring-sanitization
- 10x js/react/inconsistent-state-update
- 10x js/react/unused-or-undefined-state-property
- 1x js/redundant-operation
- 1x js/remote-property-injection
- 6x js/superfluous-trailing-arguments
- 1x js/tainted-format-string
- 8x js/trivial-conditional
- 7x js/useless-assignment-to-local
- 1x js/xss-through-dom

177 mit “Recommendation”

- 3x java/call-to-object-tostring
- 1x java/class-name-matches-super-class
- 9x java/confusing-method-signature
- 8x java/deprecated-call
- 3x java/ignored-error-status-of-call
- 1x java/inefficient-boxed-constructor
- 1x java/inefficient-empty-string-test
- 4x java/local-shadows-field
- 2x java/local-variable-is-never-read
- 44x java/missing-override-annotation
- 1x java/missing-space-in-concatenation

- 6x java/non-static-nested-class
- 3x java/uncaught-number-format-exception
- 1x java/unknown-javadoc-parameter
- 30x java/unused-parameter
- 1x java/unused-reference-type
- 1x java/useless-tostring-call
- 7x js/automatic-semicolon-insertion
- 5x js/unneeded-defensive-code
- 46x js/unused-local-variable

D 3.2.2 Übersicht über Bewertung durch den Auditor

170x Einstufung als „bestätigt“

- 3x java/call-to-object-tostring
- 1x java/class-name-matches-super-class
- 7x java/confusing-method-signature
- 8x java/deprecated-call
- 10x java/dereferenced-value-may-be-null
- 3x java/ignored-error-status-of-call
- 1x java/inconsistent-compareto-and-equals
- 1x java/inefficient-boxed-constructor
- 1x java/inefficient-empty-string-test
- 4x java/local-shadows-field
- 2x java/local-variable-is-never-read
- 44x java/missing-override-annotation
- 1x java/missing-space-in-concatenation
- 2x java/non-null-boxed-variable
- 6x java/non-static-nested-class
- 1x java/notify-instead-of-notify-all
- 5x java/random-used-once

- 1x java/relative-path-command
- 1x java/type-bound-extends-final
- 3x java/uncaught-number-format-exception
- 1x java/unknown-javadoc-parameter
- 2x java/unused-format-argument
- 19x java/unused-parameter
- 5x java/useless-null-check
- 1x java/useless-tostring-call
- 7x js/automatic-semicolon-insertion
- 10x js/react/inconsistent-state-update
- 10x js/react/unused-or-undefined-state-property
- 1x js/redundant-operation
- 6x js/superfluous-trailing-arguments
- 1x js/trivial-conditional

6x Einstufung als “verdächtig”

- 1x java/hashing-without-hashcode
- 1x java/inconsistent-equals-and-hashcode
- 1x java/unsynchronized-getter
- 1x java/unused-container
- 1x java/xss
- 1x js/log-injection

6x Einstufung als “schlechte Praxis ”

- 2x java/equals-on-unrelated-types
- 1x java/type-mismatch-access
- 2x js/comparison-between-incompatible-types
- 1x js/trivial-conditional

83x Einstufung als „kein Problem“

- 11x java/unused-parameter

- 1x java/unused-reference-type
- 1x java/weak-cryptographic-algorithm
- 1x js/incomplete-url-substring-sanitization
- 1x js/remote-property-injection
- 1x js/tainted-format-string
- 6x js/trivial-conditional
- 1x js/unbound-event-handler-receiver
- 5x js/unneeded-defensive-code
- 46x js/unused-local-variable
- 7x js/useless-assignment-to-local
- 1x js/xss
- 1x js/xss-through-dom

D 4 Semgrep

Für diese Scans kam der mitgelieferte „p/owasp-top-ten“ Regelsatz, Stand März 2022, zum Einsatz, welcher mit dem Werkzeug ausgeliefert wurde und sprachunabhängig zum Einsatz kommt. Die Scanner-Engine kam dabei in der Version 0.87.0 zum Einsatz.

Die initiale Finding-Einstufung erfolgte auf Basis der Semgrep Standard-Bewertung (error, warning). Die Sarif-Ergebnis-Dateien wurde für die Auditierung in ein einzelnes Excel-Dokument überführt, in welchem die Bewertung erfolgte. Dieses Dokument erfüllt gleichzeitig die Aufgabe des Ergebnis-Reports.

Im Folgenden werden die aus dem Scan gewonnenen Erkenntnisse zusammenfassend kurz dargestellt. Details können den jeweils angehängten Ergebnis-Dokumenten (Excel-Format) entnommen werden.

Hinweis: Die SARIF- und JSON-Exporte unterscheiden sich in der Menge der Findings im Excel-Report, da wir dort Findings im nichtproduktiven Bereich (Tests, Demos etc.), in externen Bibliotheken, sowie Duplikate im Nachgang entfernt haben.

D 4.1 BigBlueButton

Die Roh-Scans, sowie das bewertete Excel-Dokument können dem Datei-Anhang entnommen werden:

- semgrep/reports/bbb/Semgrep-bbb.xlsx
    - Komplettauflistung aller Findings inkl. Bewertung (enthält die Ergebnisse aller folgenden SARIF-Dateien, jedoch mit für den Audit ausreichender, etwas reduzierter Detaillierung)
- semgrep/raw/bbb/semgrep_bigbluebutton_owasp-top-10.json
    - unauditiertes JSON-Format aller Findings der BigBlueButton-Hauptanwendung
- semgrep/raw/bbb/semgrep_bigbluebutton_owasp-top-10.sarif
    - unauditiertes SARIF-Format aller Findings der BigBlueButton-Hauptanwendung
- semgrep/raw/bbb/semgrep_greenlight_owasp-top-10.json
    - unauditiertes JSON-Format aller Findings der Greenlight-Oberfläche
- semgrep/raw/bbb/semgrep_greenlight_owasp-top-10.sarif
    - unauditiertes SARIF-Format aller Findings der Greenlight-Oberfläche

D 4.1.1 Übersicht über Scan-Ergebnis

18 Findings wurden initial vom Werkzeug als „Error“ in folgenden Kategorien und Mengen zurückgeliefert:

- 1x java.lang.security.audit.command-injection-process-builder.command-injection-process-builder
- 14x javascript.browser.security.insecure-document-method.insecure-document-method
- 1x javascript.express.security.audit.xss.direct-response-write.direct-response-write
- 1x javascript.lang.security.detect-child-process.detect-child-process
- 1x ruby.lang.security.missing-csrf-protection.missing-csrf-protection

175 mit “Warning”:

- 2x java.lang.security.audit.crypto.use-of-sha1.use-of-sha1
- 7x javascript.browser.security.open-redirect.js-open-redirect
- 15x javascript.browser.security.raw-html-concat.raw-html-concat
- 10x javascript.browser.security.wildcard-postmessage-configuration.wildcard-postmessage-configuration
- 14x javascript.jquery.security.audit.jquery-insecure-method.jquery-insecure-method
- 13x javascript.jquery.security.audit.jquery-insecure-selector.jquery-insecure-selector
- 19x javascript.jquery.security.audit.prohibit-jquery-html.prohibit-jquery-html
- 3x javascript.lang.security.audit.path-traversal.path-join-resolve-traversal.path-join-resolve-traversal
- 7x ruby.lang.security.dangerous-exec.dangerous-exec
- 1x ruby.lang.security.dangerous-open.dangerous-open
- 4x ruby.rails.security.audit.xss.avoid-html-safe.avoid-html-safe
- 1x ruby.rails.security.audit.xss.manual-template-creation.manual-template-creation
- 3x ruby.rails.security.audit.xss.templates.dangerous-link-to.dangerous-link-to
- 43x ruby.rails.security.audit.xss.templates.unquoted-attribute.unquoted-attribute
- 12x ruby.rails.security.audit.xss.templates.var-in-href.var-in-href
- 4x typescript.react.security.audit.react-css-injection.react-css-injection

- 2x typescript.react.security.audit.react-dangerouslysetinnerhtml.react-dangerouslysetinnerhtml
- 2x typescript.react.security.audit.react-http-leak.react-http-leak
- 13x typescript.react.security.audit.react-props-injection.react-props-injection

D 4.1.2 Übersicht über Bewertung durch den Auditor

38x Einstufung als „bestätigt“

- 2x java.lang.security.audit.crypto.use-of-sha1.use-of-sha1
- 1x javascript.browser.security.raw-html-concat.raw-html-concat
- 10x javascript.browser.security.wildcard-postmessage-configuration.wildcard-postmessage-configuration
- 1x javascript.jquery.security.audit.jquery-insecure-method.jquery-insecure-method
- 3x javascript.jquery.security.audit.prohibit-jquery-html.prohibit-jquery-html
- 2x javascript.lang.security.audit.path-traversal.path-join-resolve-traversal.path-join-resolve-traversal
- 4x ruby.rails.security.audit.xss.avoid-html-safe.avoid-html-safe
- 2x typescript.react.security.audit.react-dangerouslysetinnerhtml.react-dangerouslysetinnerhtml
- 13x typescript.react.security.audit.react-props-injection.react-props-injection

155x Einstufung als „kein Problem“

- 1x java.lang.security.audit.command-injection-process-builder.command-injection-process-builder
- 14x javascript.browser.security.insecure-document-method.insecure-document-method
- 7x javascript.browser.security.open-redirect.js-open-redirect
- 14x javascript.browser.security.raw-html-concat.raw-html-concat
- 1x javascript.express.security.audit.xss.direct-response-write.direct-response-write
- 13x javascript.jquery.security.audit.jquery-insecure-method.jquery-insecure-method
- 13x javascript.jquery.security.audit.jquery-insecure-selector.jquery-insecure-selector

- 16x javascript.jquery.security.audit.prohibit-jquery-html.prohibit-jquery-html
- 1x javascript.lang.security.audit.path-traversal.path-join-resolve-traversal.path-join-resolve-traversal
- 1x javascript.lang.security.detect-child-process.detect-child-process
- 7x ruby.lang.security.dangerous-exec.dangerous-exec
- 1x ruby.lang.security.dangerous-open.dangerous-open
- 1x ruby.lang.security.missing-csrf-protection.missing-csrf-protection
- 1x ruby.rails.security.audit.xss.manual-template-creation.manual-template-creation
- 3x ruby.rails.security.audit.xss.templates.dangerous-link-to.dangerous-link-to
- 43x ruby.rails.security.audit.xss.templates.unquoted-attribute.unquoted-attribute
- 12x ruby.rails.security.audit.xss.templates.var-in-href.var-in-href
- 4x typescript.react.security.audit.react-css-injection.react-css-injection
- 2x typescript.react.security.audit.react-http-leak.react-http-leak

D 4.2 Jitsi

Die Roh-Scans, sowie das bewertete Excel-Dokument können dem Datei-Anhang entnommen werden:

- semgrep/reports/jitsi/Semgrep-jitsi.xlsx
    - Komplettauflistung aller Findings inkl. Bewertung (enthält die Ergebnisse aller folgenden SARIF-Dateien, jedoch mit für den Audit ausreichender, etwas reduzierter Detaillierung)
- semgrep/raw/jitsi/semgrep_jibri_owasp-top-10.json
    - unauditiertes JSON-Format aller Findings des Modules „jibri“
- semgrep/raw/jitsi/semgrep_jibri_owasp-top-10.sarif
    - unauditiertes SARIF-Format aller Findings des Modules „jibri“
- semgrep/raw/jitsi/semgrep_jicofo_owasp-top-10.json
    - unauditiertes JSON-Format aller Findings des Modules „jicofo“
- semgrep/raw/jitsi/semgrep_jicofo_owasp-top-10.sarif
    - unauditiertes SARIF-Format aller Findings des Modules „jicofo“
- semgrep/raw/jitsi/semgrep_jigasi_owasp-top-10.json

- unauditiertes JSON-Format aller Findings des Modules „jigasi“

- semgrep/raw/jitsi/semgrep_jigasi_owasp-top-10.sarif
  - unauditiertes SARIF-Format aller Findings des Modules „jigasi“
- semgrep/raw/jitsi/semgrep_jitsi-meet_owasp-top-10.json
  - unauditiertes JSON-Format aller Findings des Modules „jitsi-meet“
- semgrep/raw/jitsi/semgrep_jitsi-meet_owasp-top-10.sarif
  - unauditiertes SARIF-Format aller Findings des Modules „jitsi-meet“
- semgrep/raw/jitsi/semgrep_jitsi-videobridge_owasp-top-10.json
  - unauditiertes JSON-Format aller Findings des Modules „videobridge“
- semgrep/raw/jitsi/semgrep_jitsi-videobridge_owasp-top-10.sarif
  - unauditiertes SARIF-Format aller Findings des Modules „videobridge“

D 4.2.1 Übersicht über Scan-Ergebnis

5 Findings wurden initial vom Werkzeug als „Error“ in folgenden Kategorien und Mengen zurückgeliefert:

- 2x java.lang.security.audit.command-injection-process-builder.command-injection-process-builder
- 1x javascript.browser.security.insecure-document-method.insecure-document-method
- 1x javascript.browser.security.insecure-innerhtml.insecure-innerhtml
- 1x typescript.react.security.audit.react-unsanitized-property.react-unsanitized-property

43 mit “Warning”:

- 20x java.jax-rs.security.insecure-resteasy.default-resteasy-provider-abuse
- 1x java.lang.security.audit.unsafe-reflection.unsafe-reflection
- 1x javascript.browser.security.insufficient-postmessage-origin-validation.insufficient-postmessage-origin-validation
- 2x javascript.browser.security.open-redirect.js-open-redirect
- 1x javascript.browser.security.raw-html-concat.raw-html-concat
- 4x javascript.jquery.security.audit.jquery-insecure-method.jquery-insecure-method

- 2x javascript.jquery.security.audit.jquery-insecure-selector.jquery-insecure-selector
- 1x javascript.jquery.security.audit.prohibit-jquery-html.prohibit-jquery-html
- 1x javascript.lang.security.detect-non-literal-require.detect-non-literal-require
- 4x typescript.react.security.audit.react-css-injection.react-css-injection
- 1x typescript.react.security.audit.react-dangerouslysetinnerhtml.react-dangerouslysetinnerhtml
- 1x typescript.react.security.audit.react-http-leak.react-http-leak
- 4x typescript.react.security.audit.react-props-injection.react-props-injection

D 4.2.2 Übersicht über Bewertung durch den Auditor

13x Einstufung als „bestätigt“

- 1x javascript.browser.security.insufficient-postmessage-origin-validation.insufficient-postmessage-origin-validation
- 1x javascript.browser.security.open-redirect.js-open-redirect
- 1x javascript.browser.security.raw-html-concat.raw-html-concat
- 1x javascript.lang.security.detect-non-literal-require.detect-non-literal-require
- 4x typescript.react.security.audit.react-css-injection.react-css-injection
- 1x typescript.react.security.audit.react-dangerouslysetinnerhtml.react-dangerouslysetinnerhtml
- 4x typescript.react.security.audit.react-props-injection.react-props-injection

35x Einstufung als „kein Problem“

- 20x java.jax-rs.security.insecure-resteasy.default-resteasy-provider-abuse
- 2x java.lang.security.audit.command-injection-process-builder.command-injection-process-builder
- 1x java.lang.security.audit.unsafe-reflection.unsafe-reflection
- 1x javascript.browser.security.insecure-document-method.insecure-document-method
- 1x javascript.browser.security.insecure-innerhtml.insecure-innerhtml
- 1x javascript.browser.security.open-redirect.js-open-redirect
- 4x javascript.jquery.security.audit.jquery-insecure-method.jquery-insecure-method

- 2x javascript.jquery.security.audit.jquery-insecure-selector.jquery-insecure-selector
- 1x javascript.jquery.security.audit.prohibit-jquery-html.prohibit-jquery-html
- 1x typescript.react.security.audit.react-http-leak.react-http-leak
- 1x typescript.react.security.audit.react-unsanitized-property.react-unsanitized-property

D 5 TruffleHog

Diese Scans wurden mit den mit dem Werzeug mitgelieferten Standard-Regeln mit Stand vom 22.04.2022 durchgeführt. Die Scanner-Engine kam dabei in der Version 3.4.0 zum Einsatz.

Im Folgenden werden die aus dem Scan gewonnenen Erkenntnisse zusammenfassend kurz dargestellt. Details können den jeweils angehängten Ergebnis-Dokumenten (TXT-Format) entnommen werden.

D 5.1 BigBlueButton

Die Roh-Scans, sowie das TXT-Dokument können dem Datei-Anhang entnommen werden:

- trufflehog/reports/bbb/trufflehog_bbb.txt
    - TXT-Übersicht über alle Findings
- trufflehog/raw/bbb/trufflehog_bbb.json
    - JSON-Format aller Findings

Hinweis: Für folgende Module wurden keine Findings erzeugt und es sind daher auch keine Dateien im Anhang enthalten:

- bbb_api_php
- bbb_events
- bbb_install
- bbb_pads
- bbb_playback
- bbb_webhooks
- bbb_webrtc-sfu.json
- greenlight

D 5.1.1 Übersicht über Scan-Ergebnis

Es wurden 40 Findings erzeugt:

- 1x Detector Type: Gitlab
- 2x Detector Type: PrivateKey
- 26x Detector Type: Circle
- 11x Detector Type: JDBC

D 5.1.2 Übersicht über Bewertung durch den Auditor

Alle Findings wurden mit „False Positive“ bewertet.

Hinweis: Es wurden JDBC-Passwörter gefunden, welche allerdings nur im „Labs“-Bereich von BigBlueButton zum Einstz kamen und auch mehr wie Platzhakter wirkten (z.B. root/root oder bbb/secret)

D 5.2 Jitsi

Die Roh-Scans, sowie das TXT-Dokument können dem Datei-Anhang entnommen werden:

- trufflehog/reports/jitsi/trufflehog_jitsi_meet.txt
    - TXT-Übersicht über alle Findings
- trufflehog/raw/jitsi/trufflehog_jitsi_meet.json
    - JSON-Format aller Findings

Hinweis: Für folgende Module wurden keine Findings erzeugt und es sind daher auch keine Dateien im Anhang enthalten:

- jibri
- jicofo
- jigasi
- videobridge

D 5.2.1 Übersicht über Scan-Ergebnis

Es wurden 13 Findings erzeugt:

- 13x Detector Type: AmplitudeApiKey

D 5.2.2 Übersicht über Bewertung durch den Auditor

Alle Findings wurden mit „False Positive“ bewertet.

D 6 DependencyCheck

Diese Scans wurden mit den offziellen Datenbanken (u.a. NIST, NPM Public Advisories, RetireJS, Sonatype OSS Index) mit Stand vom 20.04.2022 durchgeführt. Die Scanner-Engine kam dabei in der Version 7.0.4 zum Einsatz.

Die initiale Finding-Einstufung erfolgte auf Basis der DependencyCheck Standard-Bewertung (critical, high, medium/moderate, low).

Im Folgenden werden die aus dem Scan gewonnenen Erkenntnisse zusammenfassend kurz dargestellt. Details können den jeweils angehängten Ergebnis-Dokumenten (HTML-Format) entnommen werden.

D 6.1 BigBlueButton

Die Roh-Scans, sowie das generierte HTML-Dokument können dem Datei-Anhang entnommen werden:

- dependencycheck/reports/bbb/bbb.html
    - HTML-Übersicht über alle Findings
- dependencycheck/raw/bbb/bbb.json
    - JSON-Format aller Findings
- dependencycheck/raw/bbb/bbb.sarif
    - SARIF-Format aller Findings

D 6.1.1 Übersicht über Scan-Ergebnis

132 Abhängigkeiten mit bekannten Schwachstellen (Known Vulnerabilities) wurden gefunden:

- 36x Critical
- 40x High
- 54x Medium/Moderate
- 2x Low

D 6.1.2 Übersicht über Bewertung durch den Auditor

Da die vom Tool gewonnen Informationen auf Basis bekannter und verbreiteter Datenbanken beruhen, nehmen wir alle Findings als „bestätigt“ an. Eine feingranulare Bewertung (z.B. durch Nichtverwendung vulnerabler Bibliotheksanteile) kann aus unserer Erfahrung nur durch Projekt-Insider erfolgen.

D 6.2 Jitsi

Die Roh-Scans, sowie das generierte HTML-Dokument können dem Datei-Anhang entnommen werden:

- dependencycheck/reports/jitsi/jitsi.html
    - HTML-Übersicht über alle Findings
- dependencycheck/raw/jitsi/jitsi.json
    - JSON-Format aller Findings
- dependencycheck/raw/jitsi/jitsi.sarif
    - SARIF-Format aller Findings

D 6.2.1 Übersicht über Scan-Ergebnis

65 Abhängigkeiten mit bekannten Schwachstellen (Known Vulnerabilities) wurden gefunden:

- 9x Critical
- 22x High
- 34x Medium/Moderate

D 6.2.2 Übersicht über Bewertung durch den Auditor

Da die vom Tool gewonnen Informationen auf Basis bekannter und verbreiteter Datenbanken beruhen, nehmen wir alle Findings als „bestätigt“ an. Eine feingranulare Bewertung (z.B. durch Nichtverwendung vulnerabler Bibliotheksanteile) kann aus unserer Erfahrung nur durch Projekt-Insider erfolgen.

E. Testergebnisse SAST-Analysen detailliert

Neben den direkt aus den automatischen, werkzeugbasierten Analysen resultierenden Ergebnissen (Details s. Kapitel D) werden hier Beobachtungen aufgelistet, welche sich durch besondere Kritikalität auszeichneten bzw. durch unsere manuellen Code-Analysen aufgefunden wurden.

E 1 BigBlueButton

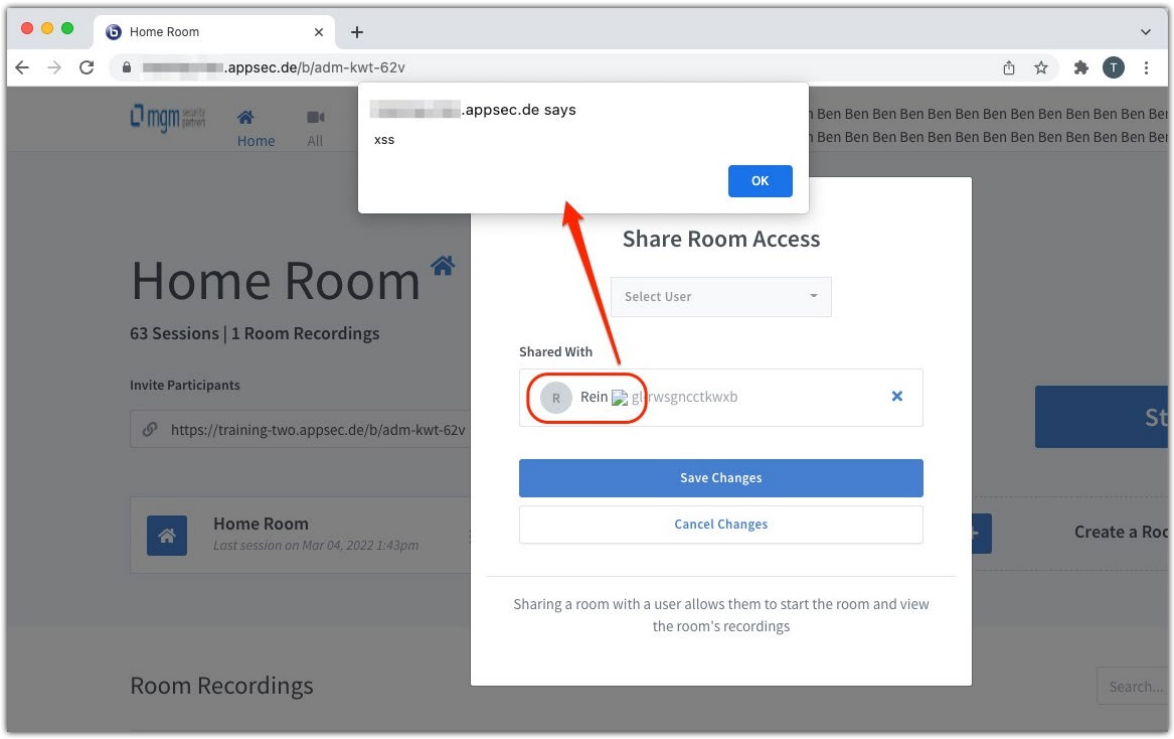
E 1.1 Cross-Site-Scripting

E 1.1.1 BBB: Stored-Cross-Site-Scripting-Schwachstelle in Bigbluebutton/Greenlight [hoch]

Beobachtung

Das Dialogfeld "Share Room Access" wurde als anfällig für Stored-Cross-Site-Scripting befunden.

Ein Angreifer kann JavaScript-Code in seinen Benutzernamen einfügen. Dieser Code wird im Browser des Opfers im Dialogfeld "Share Room Access" ausgeführt, wenn das Opfer zuvor den Zugang zum Raum mit dem Angreifer geteilt hat.



Schritte zur Reproduktion der Schwachstelle:

1. Benutzer A (Opfer) teilt den Raumzugang mit Benutzer B (Angreifer).
2. Benutzer B aktualisiert seinen Benutzernamen, um einen XSS-Payload zu injizieren. Für den obigen Proof-of-Concept wurde der Payload "`Rein <img src=x onerror=alert('xss')>`" verwendet.
3. Benutzer A öffnet erneut den Dialog "Raumzugang freigeben" des Raums. Der von Benutzer B injizierte Payload wird ausgeführt.

Anfälliger Code:

— Senke: https://github.com/bigbluebutton/greenlight/blob/release-2.11.1/app/assets/javascripts/room.js#L347

Siehe auch 0 (durch uns gemeldete CVE-2022-26497).

Ausnutzbarkeit

Um die Schwachstelle ausnutzen zu können, muss ein authentifizierter Benutzer lediglich eine entsprechende Payload in seinem Benutzernamen eingeben und diesen speichern. Dadurch kann JavaScript-Code dauerhaft in die Applikation eingeschleust werden. Bei jedem Benutzer, der diese Informationen abruft, wird dann der eingeschleuste Code ausgeführt.

Mit dieser Schwachstelle kann ein Angreifer außerdem leicht den Vertrauenskontext von BigBlueButton ausnutzen, da er mittels JavaScript und HTML den gesamten Inhalt der Seite manipulieren kann. Dies kann dazu verwendet werden, falsche Login-Dialoge oder ähnliches anzuzeigen und darauf zu hoffen, dass ein Benutzer Passwörter eingibt oder andere Informationen preisgibt, die der Angreifer dann für weitere Angriffe verwenden kann. Im schlimmsten Fall würde das dem Angreifer ermöglichen, die Identität des Benutzers zu übernehmen und den Account zu missbrauchen.

Ein Angreifer kann auch gezielt Falschinformationen in der Seite einbauen, um das Unternehmen zu schädigen.

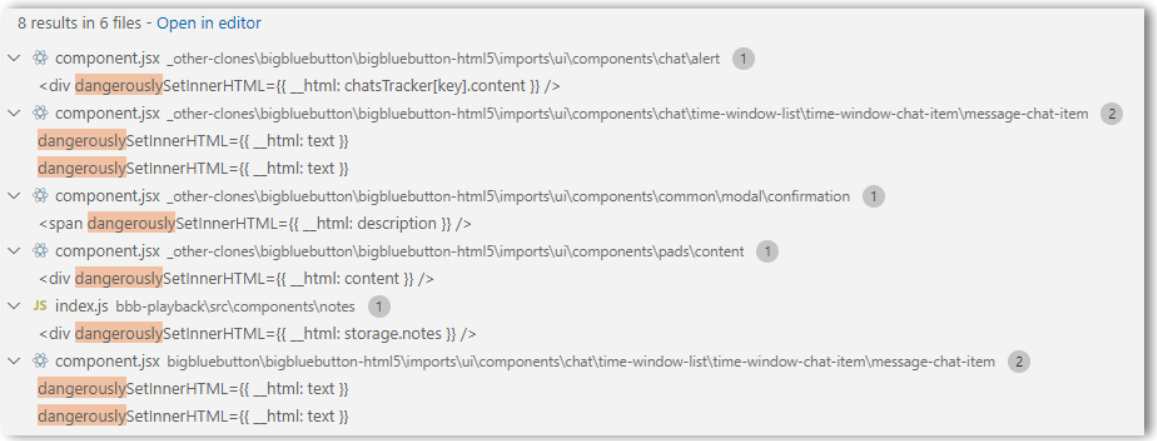
Maßnahme

Es sollten keine Benutzer-Eingaben ungefiltert bzw. ohne entsprechendes Encoding in der Response zurückgegeben werden.

E 1.1.2 BBB: Verwendung `dangerouslySetInnerHTML` als mögliche XSS-Quelle [info]

Beobachtung

Es ist aufgefallen, dass im react-Frontend auf die XSS-gefährdete `dangerouslySetInnerHTML` zurückgegriffen wird. Folgende Stellen sind betroffen:



Ausnutzbarkeit

Diese Stellen sind im untersuchten Code kein Problem, da keine dynamischen Daten in die Ausgaben einfließen. Sie können sich jedoch durch kleine unachtsame Änderungen leicht zu einer Schwachstelle entwickeln (wie in C1.23 CVE-2020-12113 in der Vergangenheit bereits geschehen). Das Problem hierbei ist, dass der Entwickler oft an einer ganz anderen Stelle erweitert (z.B. Anhängen von Nutzerinput an den Variableninhalt im Server-Code) und er nach unserer Erfahrung häufig nicht erkennt, dass er zusätzlich ins Ausgabetemplate zu schauen und dort den Typ der Ausgabe zu prüfen/ändern hat.

Maßnahme

Idealerweise verzichtet man vollständig auf die Verwendung von `dangerouslySetInnerHTML`. Dies geht in der Praxis mit Mehraufwand bei der Umsetzung einher, sollte aber aus Sicherheitssicht erwogen werden.

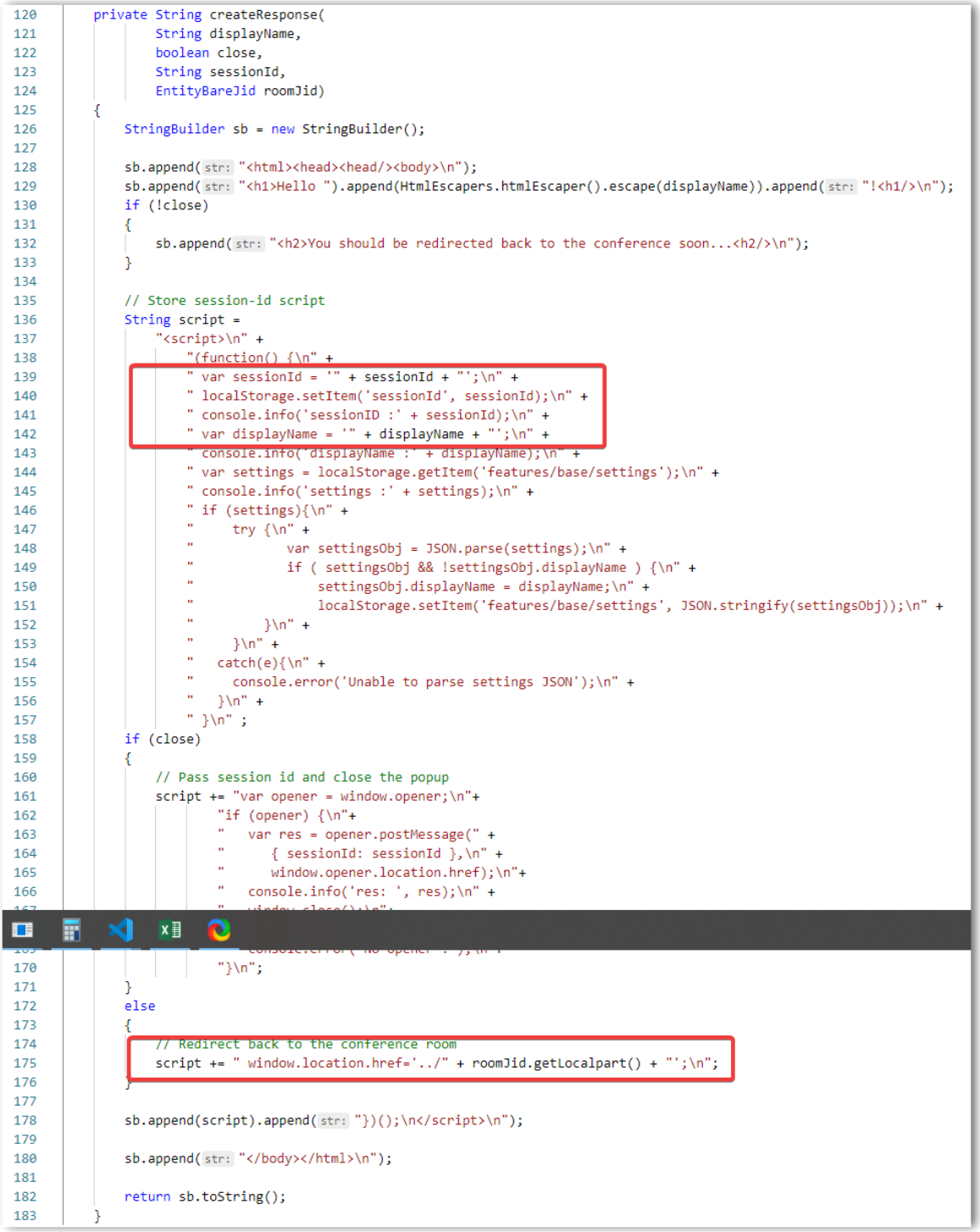
E 2 Jitsi

E 2.1 Cross-Site-Scripting

E 2.1.1 Jitsi: Verdacht auf Cross-Site-Scripting im Shibboleth-Login [info]

Beobachtung

In der Klasse `ShibbolethLogin.java` wurde eine Code-Stelle (Methode `createResponse`) identifiziert, welche `<script>`-Inhalte für die Ausgabe in die HTML-Login-Seite aus statischen und dynamischen Inhalten durch Konkatenation erstellt:



```
120    private String createResponse(
121            String displayName,
122            boolean close,
123            String sessionId,
124            EntityBareJid roomJid)
125    {
126        StringBuilder sb = new StringBuilder();
127
128        sb.append( str: "<html><head><head/><body>\n");
129        sb.append( str: "<h1>Hello ").append(HtmlEscapers.htmlEscaper().escape(displayName)).append( str: "!<h1/>\n");
130        if (!close)
131        {
132            sb.append( str: "<h2>You should be redirected back to the conference soon...<h2/>\n");
133        }
134
135        // Store session-id script
136        String script =
137            "<script>\n" +
138                "(function() {\n" +
139                " var sessionId = '" + sessionId + "';\n" +
140                " localStorage.setItem('sessionId', sessionId);\n" +
141                " console.info('sessionID :' + sessionId);\n" +
142                " var displayName = '" + displayName + "';\n" +
143                " console.info('displayName :' + displayName);\n" +
144                " var settings = localStorage.getItem('features/base/settings');\n" +
145                " console.info('settings :' + settings);\n" +
146                " if (settings){\n" +
147                "     try {\n" +
148                "             var settingsObj = JSON.parse(settings);\n" +
149                "             if ( settingsObj && !settingsObj.displayName ) {\n" +
150                "                 settingsObj.displayName = displayName;\n" +
151                "                 localStorage.setItem('features/base/settings', JSON.stringify(settingsObj));\n" +
152                "         }\n" +
153                "     }\n" +
154                "   catch(e){\n" +
155                "     console.error('Unable to parse settings JSON');\n" +
156                "   }\n" +
157                " }\n" ;
158        if (close)
159        {
160            // Pass session id and close the popup
161            script += "var opener = window.opener;\n"+
162                    "if (opener) {\n"+
163                    "   var res = opener.postMessage(" +
164                    "      { sessionId: sessionId },\n" +
165                    "      window.opener.location.href);\n"+
166                    "   console.info('res: ', res);\n" +
170                    "}\n";
171        }
172        else
173        {
174            // Redirect back to the conference room
175            script += " window.location.href='../" + roomJid.getLocalpart() + "';\n";
176        }
177
178        sb.append(script).append( str: "})();\n</script>\n");
179
180        sb.append( str: "</body></html>\n");
181
182        return sb.toString();
183    }
```

An den beiden markierten Stellen wird der Inhalt von Variablen (`sessionId`, `displayName`, `roomJid.getLocalpart()`) gemischt mit statischen Inhalten an einen `StringBuffer` angehängt (über die Verwendung eines temporären `String`s, was zudem Fragen der generellen Code-Qualität an dieser Stelle aufwirft).

Ausnutzbarkeit

Im vorliegenden Fall konnte eine direkt Einflussnahme von außen nicht nachgewiesen werden, jedoch kann es auch nicht 100%ig ausgeschlossen werden. Eine dynamische Prüfung war zum Zeitpunkt der Analyse leider nicht möglich, da das Setup von Shibboleth nicht Bestandteil der DAST-Analysen war.

Hinweis: In einem passenden Setup kann dies leicht nachgeholt werden.

Maßnahme

Grundsätzlich sollte diese Form von serverseitiger HTML-Konkatenation vermieden werden.

E 2.1.2 Jitsi: Verwendung `dangerouslySetInnerHTML` als mögliche XSS-Quelle [info]

Beobachtung

Es ist aufgefallen, dass im react-Frontend auf die XSS-gefährdete `dangerouslySetInnerHTML` zurückgegriffen wird. Folgende Stelle ist betroffen:

Ausnutzbarkeit

Diese Stelle ist im untersuchten Code kein Problem, da keine dynamischen Daten in die Ausgaben einfließen. Sie können sich jedoch durch kleine unachtsame Änderungen leicht zu einer Schwachstelle entwickeln. Das Problem hierbei ist, dass der Entwickler oft an einer ganz anderen Stelle erweitert (z.B. Anhängen von Nutzerinput an den Variableninhalt im Server-Code) und er nach unserer Erfahrung häufig nicht erkennt, dass er zusätzlich ins Ausgabetemplate zu schauen und dort den Typ der Ausgabe zu prüfen/ändern hat.

Maßnahme

Idealerweise verzichtet man vollständig auf die Verwendung von `dangerouslySetInnerHTML`. Dies geht in der Praxis mit Mehraufwand bei der Umsetzung einher, sollte aber aus Sicherheitssicht erwogen werden.

F. Testergebnisse DAST-Analysen detailliert

In diesen Kapitel werden Beobachtungen aufgelistet, welche durch manuelle DAST- und IAST-Ansätze aufgefunden wurden.

F 1 BigBlueButton

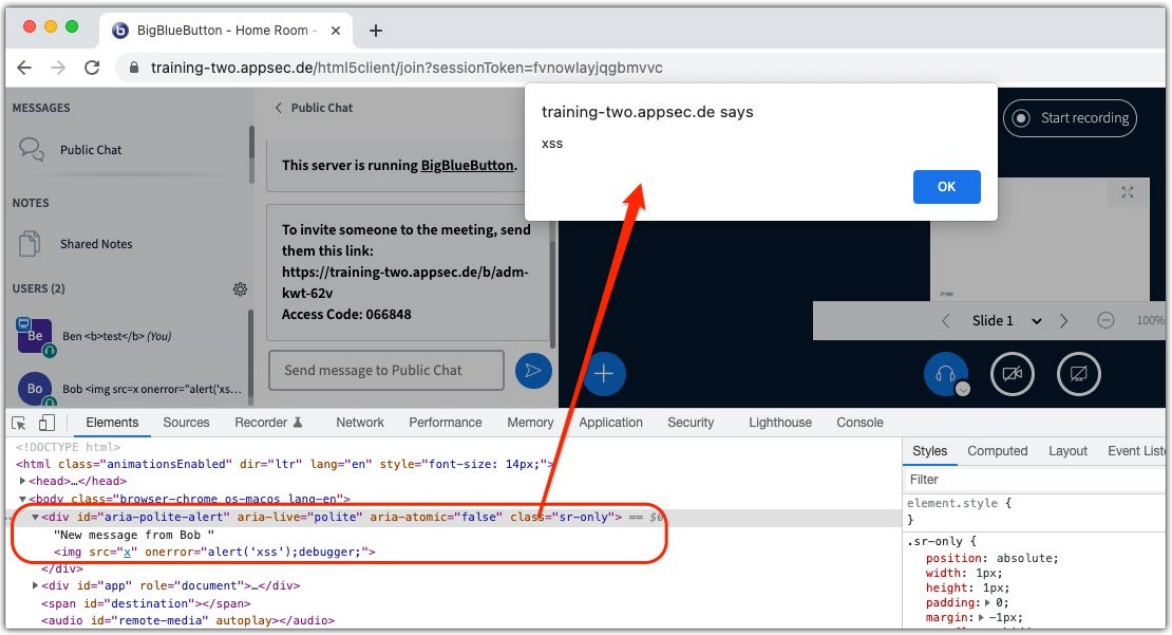
F 1.1 Cross-Site-Scripting

F 1.1.1 BBB: Stored-XSS-Schwachstelle in Bigbluebutton/bigbluebutton-html5 [hoch]

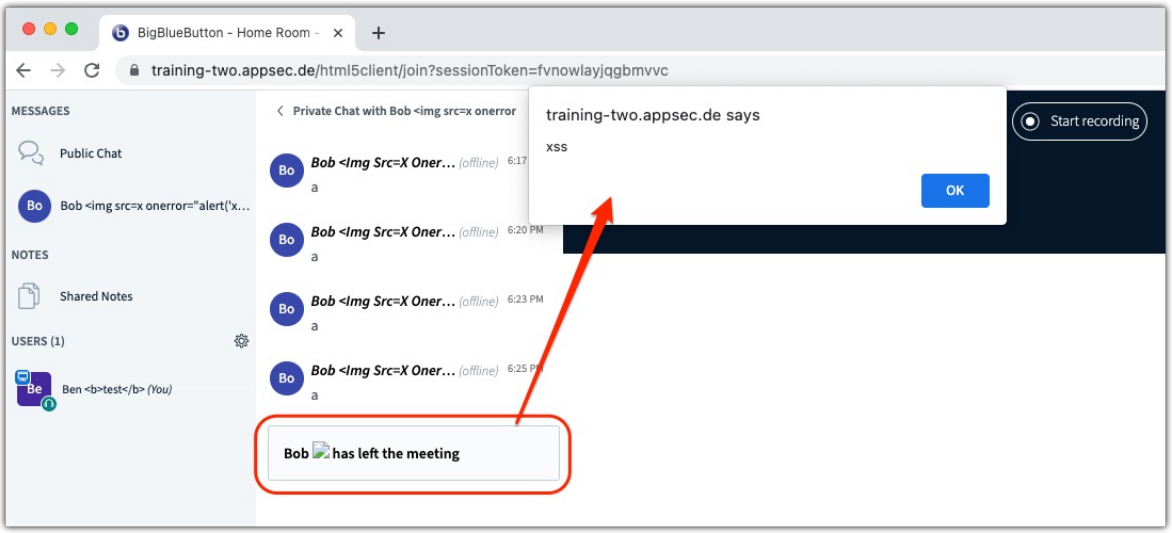
Beobachtung

Die "Private Chat"-Funktionalität in Bigbluebutton Meeting wurde als anfällig für Stored-Cross-Site-Scripting befunden.

Dies ist möglich, wenn ein Angreifer HTML/JavaScript-Code in seinen Benutzernamen einfügt. Der Code im Benutzernamen wird im Browser des Opfers jedes Mal ausgeführt, wenn der Angreifer eine private Nachricht an das Opfer sendet.



Wenn der Angreifer die Besprechung verlässt, ist auch die Benachrichtigung über sein Verlassen anfällig für XSS. Der folgende Screenshot veranschaulicht das Problem.



Schritte zur Reproduktion der Schwachstelle:

1. Benutzer A (Opfer) und Benutzer B (Angreifer) nehmen an der gleichen Besprechung teil

2. Benutzer B fügt einen XSS-Payload in seinen Benutzernamen ein. Für den obigen Proof-of-Concept wurde der Payload "`Bob <img src=x onerror="alert('xss');debugger;">`" verwendet.

3. Benutzer B startet einen privaten Chat mit Benutzer A. Nach jeder Nachricht von Benutzer B an Benutzer A wird der in den Benutzernamen von Benutzer B injizierte Payload ausgeführt.

4. Nachdem Benutzer B den Raum verlassen hat, wird der Payload erneut ausgeführt, wenn Benutzer A die Benachrichtigung über das Verlassen von B erhält.

Anfälliger Code:

— Senke [1]: https://github.com/bigbluebutton/bigbluebutton/blob/v2.4.4/bigbluebutton-html5/imports/ui/components/nav-bar/component.jsx#L201 (Benutzername wird nicht escaped).

— Senke [2]: https://github.com/bigbluebutton/bigbluebutton/blob/v2.4.4/bigbluebutton-html5/imports/ui/components/chat/container.jsx#L202

Siehe auch C1.1 (durch uns gemeldete CVE-2022-27238).

Ausnutzbarkeit

Siehe E1.1.1.

Maßnahme

Siehe E1.1.1.

F 1.2 Open Redirect

F 1.2.1 BBB: Open Redirect (Selbstangriff) in Bigbluebutton/Greenlight [info]

Beobachtung

Die Greenlight-Anwendung verwendet die "`redirect_back`"-Methode bei mehreren Formularen nach der Bearbeitung der Anfrage, zum Beispiel:

— POST /admins/approve/:user_uid

— POST /admins/ban/:benutzer_uid

— POST /:room_uid/remove_presenstation

Die "`redirect_back`"-Methode des Rails-Frameworks leitet den Benutzer zu der Seite um, von der die Anfrage ausging (angegeben durch den Referrer-Header).

Dieses Verhalten ist an sich nicht ausnutzbar, da der Referrer-Header in einem Cross-Site-Szenario nicht manipuliert werden kann. Wenn diese Endpunkte jedoch auch die GET-Methode akzeptieren (was in der getesteten Anwendung nicht der Fall ist), könnte die offene Umleitung ausgenutzt werden.

Ausnutzbarkeit

Dieses Verhalten ist an sich nicht ausnutzbar, da der Referrer-Header in einem Cross-Site-Szenario nicht manipuliert werden kann. Wenn diese Endpunkte jedoch auch die GET-Methode akzeptieren (was in der getesteten Anwendung derzeit nicht der Fall ist), könnte ein Open Redirect ausgenutzt werden.

Maßnahme

Die "`redirect_back`"-Methode sollte sorgfältig eingesetzt werden. Auf die Verwendung bei GET-Requests ist hierbei zu verzichten.

F 1.3 Path-Traversal

F 1.3.1 BBB: Path Traversal in Bigbluebutton/Greenlight [info]

Beobachtung

In `Admin > Site settings` ermöglicht die Anwendung das Rendern eines beliebigen, durch eine Benutzereingabe festgelegten Scripts. Zum Beispiel:

```
https://<site>/b/admins/site_settings?tab=settings
```

Der Abfrageparameter "`tab`" in der URL wird verwendet, um den Namen des einzuschließenden Skripts anzugeben.

Der folgende Screenshot zeigt den anfälligen Quellcode (https://github.com/bigbluebutton/greenlight/blob/release-2.11.1/app/views/admins/components/_setting_view.html.erb)

Ausnutzbarkeit

Die Einbindung lokaler Dateien könnte möglicherweise zum Lesen lokaler Dateien oder zu einer Remote-Code-Execution führen. Die Ausnutzbarkeit dieser Schwachstelle ist jedoch sehr begrenzt, da das Rails-Framework über eingebaute Maßnahmen zur Verhinderung von Path-Traversal-Angriffen verfügt:

- nur Dateien im Verzeichnis "views" können eingebunden werden
- der Dateiname des Scripts muss das richtige Format haben (`_<Dateiname>.html.erb`)

Maßnahme

Nutzereingaben sollten nicht ungeprüft verwendet werden, um lokale Dateien einzubetten.

F 1.4 Fehlende Authentisierung

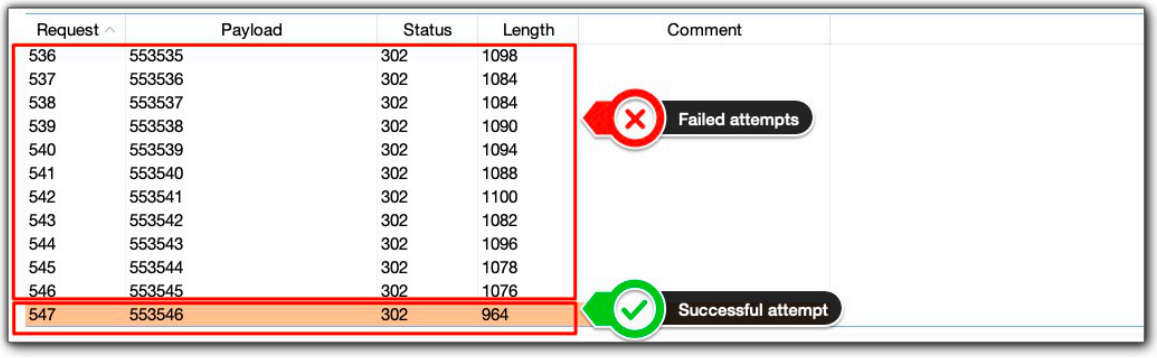
F 1.4.1 BBB: Brute-Force auf Zugriffcode für Räume in Bigbluebutton/Greenlight möglich [niedrig]

Hinweis: Bereits mit CVE-2020-29042 gemeldet und nicht korrigiert (s. C1.5).

Beobachtung

Räume können so konfiguriert werden, dass die Teilnehmer einen Zugangscode eingeben müssen, bevor sie dem Raum beitreten dürfen. Der Endpunkt für die Überprüfung des Zugangscodes verfügt über keinen Schutz gegen Automatisierung. Angreifer könnten den Zugangscode (6-stellige Ganzzahl) mittels Brute-Force Angriff erraten.

Wir haben ein Tool (Burp Intruder) verwendet, um den Schutz der Anwendung gegen Brute-Force bei der Prüfung des Zugangscodes für den Raum zu analysieren. Während der Analyse wurden über 500 Versuche, einen Raum mit ungültigen Zugangscodes zu betreten, gesendet. Anschließend gelang noch ein Versuch mit dem gültigen Code. Die erfolgreiche Anfrage (mit dem korrekten Zugangscode) ist auf dem folgenden Screenshot zu sehen, mit der kürzeren Antwortlänge (964 Bytes im Vergleich zu den 1000+ Bytes der ungültigen), für die Anfrage #547.



Ausnutzbarkeit

Ein Angreifer kann mit relativ geringem Zeitaufwand den Zugriffscode für einen Raum erraten und so den Raum im Erfolgsfall als legitimer Nutzer betreten. Da ein Angreifer auch die ID des Raumes kennen muss, wird diese Schwachstelle lediglich mit [niedrig] bewertet.

Maßnahme

Anstelle eines autoamtisch generierten 6-stelligen Wertes, sollte die Anwendung einem Benutzer die Möglichkeit bieten, auch längere und komplexere Zugriffscodes eingeben zu können.

F 1.5 Bekanntgabe von unterschiedlichen Systeminformationen

F 1.5.1 BBB: Bekanntgabe von Verifikationstoken-Hashs in Bigbluebutton/Greenlight [info]

Hinweis: Bereits mit CVE-2020-29043 gemeldet und nicht vollständig korrigiert (s. C1.4).

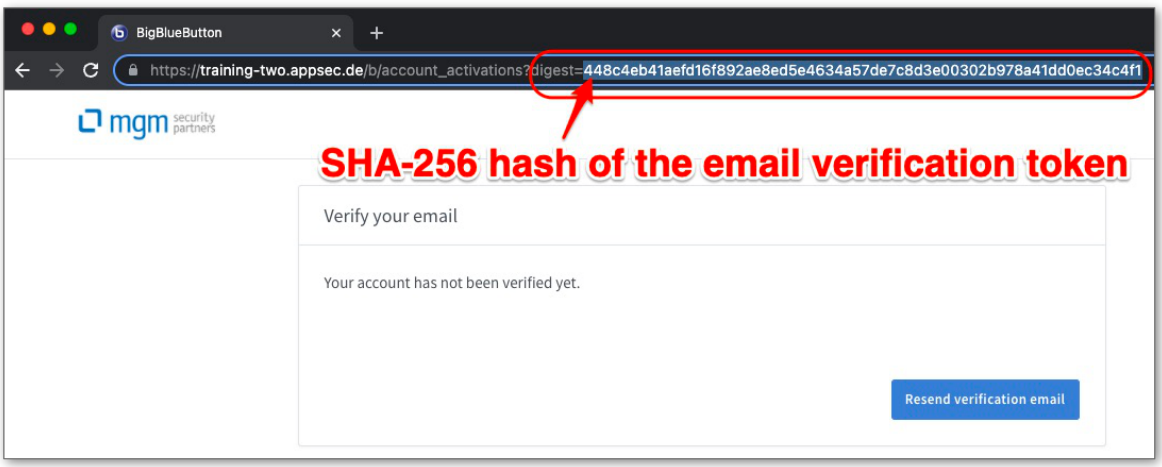
Beobachtung

Greenlight kann so konfiguriert werden, dass neu registrierte Nutzer ihre E-Mail-Adresse verifizieren müssen, bevor sie die Anwendung nutzen können. Um ihre E-Mail-Adresse zu verifizieren, müssen die Benutzer eine URL öffnen, die ein geheimes Token enthält, das an ihre E-Mail gesendet wird. Die URL sieht wie folgt aus:

```
/b/account_activations/edit?token=<VerificationToken>
```

Wenn der Benutzer nach der Anmeldung seine E-Mail-Adresse nicht verifiziert hat, wird er zu einer URL der folgenden Form weitergeleitet:

```
/b/account_activations?digest=<Hash>
```



Es konnte bestätigt werden, dass der `<Hash>` in der späteren URL ein SHA-256-Hash des `<VerificationToken>` ist.

Ausnutzbarkeit

Ein Angreifer könnte den offengelegten Hash erraten (Offline-cracking von SHA-256), um den <VerificationToken> zu erhalten. Bei Erfolg könnten man den `<VerificationToken>` verwenden, um die Inhaberschaft der angegebenen E-Mail-Adresse zu überprüfen, ohne Zugriff auf das Postfach haben zu müssen.

Da das `<VerificationToken>` lang und komplex ist (das Token besteht aus 22 Zeichen, Kleinbuchstaben + Großbuchstaben + Zahlen + Unterstrichen), ist die Ausnutzbarkeit dieser Schwachstelle begrenzt.

Maßnahme

Token sollten nicht unnötigerweise exponiert werden.

G. Anhänge

Folgende Anhänge wurden als Teil dieses Berichtes mitgeliefert:

- Ordner: burp
  - Arbeitsprotokolle Burp-Analysen
- Ordner: checkmarx
  - Reports und RAW-Exporte vom Werzeug Checkmarx SAST (Details siehe D2)
- Ordner: codeql
  - Reports und RAW-Exporte vom Werzeug CodeQL (Details siehe D3)
- Ordner: contrast
  - Findings-Export Contrast-IAST
- Ordner: dependencycheck
  - Reports und RAW-Exporte vom Werzeug OWASP DependencyCheck (Details siehe D6)
- Ordner: fortify
  - Reports und RAW-Exporte vom Werzeug Microfocus Fortify (Details siehe D1)
- Ordner: semgrep
  - Reports und RAW-Exporte vom Werzeug Semgrep (Details siehe D4)
- Ordner: trufflehog
  - Reports und RAW-Exporte vom Werzeug TruffleHog (Details siehe D5)

H. Referenzen

/1/ Input- und Output-Datenvalidierung in Webanwendungen
https://cwe.mitre.org/data/definitions/79.html

/2/ Übersicht zu Datenvalidierung
https://www.owasp.org/index.php/Data_Validation

/3/ Cheat Sheet zu Datenvalidierung
https://www.owasp.org/index.php/Input_Validation_Cheat_Sheet

/4/ Übersicht zur XSS-Prävention
https://www.owasp.org/index.php/XSS_%28Cross_Site_Scripting%29_Prevention_Cheat_Sheet

/5/ Unvalidierte Redirects und Forwards
https://www.owasp.org/index.php/Unvalidated_Redirects_and_Forwards_Cheat_Sheet

/6/ Übersicht zu SQL-Injection
https://www.owasp.org/index.php/SQL_Injection

/7/ Übersicht zu Blind-SQL-Injection
https://www.owasp.org/index.php/Blind_SQL_Injection

/8/ SQL-Injection-Prävention
https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet

/9/ Best Practice Maßnahmen zur Auditierung und Härtung von Datenbanken
https://benchmarks.cisecurity.org/downloads/browse/index.cfm?category=benchmarks.servers.database

/10/ System-Kommandos aufrufen
https://cwe.mitre.org/data/definitions/78.html

/11/ Path-Traversal
https://www.owasp.org/index.php/Path_Traversal

/12/ Buffer-Overflow-Angriff
https://www.owasp.org/index.php/Buffer_overflow_attack

/13/ Format-String-Angriff
https://www.owasp.org/index.php/Format_string_attack

/14/ Benutzer-Enumeration
https://www.owasp.org/index.php/Testing_for_user_enumeration_(OWASP-AT-002)

/15/ OWASP - Blocking Brute Force Attacks
https://www.owasp.org/index.php/Blocking_Brute_Force_Attacks

/16/ Länge und Komplexität von Passwörtern
https://www.owasp.org/index.php/Authentication_Cheat_Sheet#Password_Complexity

/17/ Übersicht zu Authentication
https://www.owasp.org/index.php/Authentication_Cheat_Sheet

/18/ Übersicht zu Session-Management
https://www.owasp.org/index.php/Session_Management_Cheat_Sheet

/19/ Session-Fixation-Schwachstelle
http://www.acros.si/papers/session_fixation.pdf

/20/ Einführung in CSRF bzw. Session Riding
https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)

/21/ CSRF-Prävention
https://www.owasp.org/index.php/Cross-Site_Request_Forgery_%28CSRF%29_Prevention_Cheat_Sheet

/22/ Privilegienerweiterung
https://cwe.mitre.org/data/definitions/269.html

/23/ Zugriffskontrolle
https://www.owasp.org/index.php/Access_Control_Cheat_Sheet

/24/ Sichere Passwort-Verwaltung
https://www.owasp.org/index.php/Password_Storage_Cheat_Sheet

/25/ Sichere Datenspeicherung
https://www.owasp.org/index.php/Cryptographic_Storage_Cheat_Sheet

/26/ Best Practices for a Secure “Forgot Password” Feature
https://www.owasp.org/index.php/Forgot_Password_Cheat_Sheet

/27/ Fehlerhafte Benutzer-Authentisierung und Session-Verwaltung
https://www.owasp.org/index.php/Broken_Authentication_and_Session_Management

/28/ Denial of Service
https://www.owasp.org/index.php/Denial_of_Service

/29/ CAPTCHA: Telling Humans and Computers Apart Automatically
http://www.captcha.net/

/30/ SSL Best Practices
https://www.ssllabs.com/projects/best-practices/index.html

/31/ Übersicht von öffentlich bekannten Schwachstellen
https://nvd.nist.gov/

/32/ Secure Coding of CORS
http://www.andlabs.org/html5/rejectCOR.php

/33/ Informationen über HSTS
http://blog.securenet.de/2012/11/02/ssl-stripping-die-ignorierte-gefahr/

/34/ CWE-77: Improper Neutralization of Special Elements used in a Command ('Command Injection')
https://cwe.mitre.org/data/definitions/77.html

/35/ OWASP Content Security Policy
https://www.owasp.org/index.php/Content_Security_Policy

/36/ OWASP Clickjacking
https://www.owasp.org/index.php/Clickjacking

/37/ OWASP HTML5 Security Cheat Sheet
https://www.owasp.org/index.php/HTML5_Security_Cheat_Sheet

/38/ OWASP HTTP Headers
https://www.owasp.org/index.php/List_of_useful_HTTP_headers

/39/ Unrestricted Upload of File with Dangerous Type
https://cwe.mitre.org/data/definitions/434.html

/40/ Unrestricted File Upload
https://www.owasp.org/index.php/Unrestricted_File_Upload

/41/ OWASP TLS Cheat Sheet
https://www.owasp.org/index.php/Transport_Layer_Protection_Cheat_Sheet

/42/ Sichere Passwortablage
https://paragonie.com/blog/2016/02/how-safely-store-password-in-2016