

LEANOFY

PRODUKTDATENBLATT · WM-AUDIT 1.0.0 · WINDOWS · MACOS · LINUX

Security-Audit für das webMethods API Gateway

wm-audit liest den Export Ihres Gateways und zeigt, wo die **tatsächliche Konfiguration** Schnittstellen offenlässt: fehlende Identifikation, anonymer Zugriff, unverschlüsselte Wege. Lokal und ohne Netzwerk, ohne eine einzige Anfrage an Ihre APIs – und ohne Passwörter und Schlüssel im Export zu öffnen.

NUTZEN · BEISPIEL: 100 APIS IM GATEWAY

1.000

Einzelprüfungen

10 Prüfregele je API, in einem Lauf über den ganzen Katalog

0,01 s

für 100 APIs

gemessen: 556 Exportdateien, Mac mit Apple Silicon; 1.000 APIs in 0,15 s

1 statt 100

Durchgänge

von Hand je API: öffnen, Reiter Policies, bis zu sieben Policy-Stufen prüfen

Beispielrechnung: Bei angenommenen 5 Minuten je API sind 100 APIs von Hand gut 8 Stunden Klickarbeit – nach jeder Änderung von vorn. wm-audit wiederholt die Prüfung in jeder Pipeline (--junit, --fail-on) und liefert die Aufgabenliste mit Anleitung gleich mit.



ECHTE AUSGABE · AUSZUG

```
$ wm-audit gateway-export.zip
wm-audit 1.0.0 · gateway-export.zip (zip)
10 APIs gelesen (10 eindeutig) · 1 Dateien übersprungen · 2 gesperrte Einträge nicht geöffnet
CRITICAL belegt gw.anonymous_allowed AnonymAPI 1.0
HIGH belegt gw.identify_or OderAPI 1.0
HIGH belegt gw.plaintext_backend KlartextAPI 1.0
...
```

PRÜFREGELEN

Was wm-audit im Export findet

CRITICAL	Identify-Policy lässt anonymen Zugriff zu allowAnonymous hebt die Identifikation auf	OWASP API2
HIGH	API ohne Identify-and-Authorize-Policy Das Gateway reicht jeden Aufrufer durch	OWASP API2
HIGH	Identifikationsverfahren ODER-verknüpft Der API-Key allein genügt, wo Key und Token gemeint waren	OWASP API2
HIGH	Gateway nimmt unverschlüsseltes HTTP an Entry-Protocol-Policy erlaubt HTTP	OWASP API8
HIGH	Routing auf ein Klartext-Backend Routing-Ziel inklusive Alias-Auflösung	OWASP API8
MEDIUM	Nur Anwendungs-, keine Benutzeridentität Benutzerbezogene Prüfung liegt allein im Backend	OWASP API2
MEDIUM	Security-Header gehen ans Backend passSecurityHeaders reicht Authorization weiter	OWASP API8
MEDIUM	Keine Mengen- oder Größenbegrenzung Keine Traffic-Policy auf API-Ebene	OWASP API4
LOW	Kontingent über einen Tag oder länger Bremst laut IBM-Doku keine Lastspitzen	OWASP API4
LOW	Anfragen werden protokolliert Log-Invocation kann Nutzdaten mitschreiben	OWASP API8

FÜR IHREN FREIGABEPROZESS

Sicherheitseigenschaften — jede mit Testnachweis



Kein Netzwerk

Keine Netzwerk- oder TLS-Bibliotheken im Programm — per Bann-Liste in der Lieferkettenprüfung ausgeschlossen.



Öffnet keine Passwörter und Schlüssel

PassmanData, Keystore, Truststore und Kerberos werden nie geöffnet, nur gezählt. Passwörter in Adressen werden im Bericht geschwärzt.



Nur lesend

Der Export bleibt unverändert, ein ZIP wird nie entpackt. Geschrieben wird nur die angegebene Berichtsdatei; es startet keine Prozesse.



Speichersicher

Rust ohne unsafe-Code. Abbrüche über unwrap, expect und panic sind per Prüfregel ausgeschlossen, Überlaufprüfung auch im Release.



Robust gegen manipulierte Exporte

Grenzen gegen ZIP- und JSON-Bomben (32 MiB je Datei, 512 MiB gesamt, Tiefe 100). Doppelte ZIP-Einträge werden abgelehnt, Symlinks nicht verfolgt.



Transparent

Jede übersprungene Datei steht mit Grund im Bericht, dazu der SHA-256 der geprüften Datei. Keine Benutzerpfade im Bericht.



Deterministisch

Gleicher Export, byte-gleicher Bericht, stabile Befund-IDs — Ergebnisse lassen sich vergleichen und archivieren.



Gegen eine Referenz geprüft

Identisch mit der Referenzimplementierung: 15 Golden-Fälle, 10.000 zufällig erzeugte Exporte, öffentliche Software-AG-Exporte.

LIEFERKETTE UND BUILD

23

Crates

vollständige Abhängigkeiten, nur crates.io

3

Lizenzen

MIT · Apache-2.0 · Zlib — kein Copyleft

0

bekannte Schwachstellen

cargo-audit gegen RustSec, 13.10.2026

1:1

reproduzierbar

zwei Builds bitgleich, SBOM als CycloneDX

TECHNISCHE DATEN

EINGABE	Export des webMethods API Gateway — ZIP-Datei oder entpackter Ordner
AUSGABE	HTML-Bericht mit Aufgabenliste und Lastrechner; Text, JSON oder JUnit/XML
CI / CD	--junit für Jenkins, GitLab & Co.; --fail-on: Exit-Code 1 ab Schwere
PLATTFORMEN	Windows, macOS (Apple Silicon, Intel), Linux — je ein Programm, keine Installation
LIZENZ	Apache-2.0; Quelltext öffentlich: github.com/cehlert9999/wm-audit
NACHPRÜFEN	./prüfen.sh: alle Prüfungen und der Build in einem Lauf
DOKUMENTATION	leanofy.de/wm-audit/dokumentation

STATISCH NICHT ENTSCHEIDBAR

Was kein Export beantworten kann

wm-audit benennt diese Fragen im Bericht, statt sie zu bewerten:

- ? Wird ein erfundener API-Key abgewiesen?
- ? Wird die JWT-Signatur wirklich geprüft?
- ? Erreicht ein Aufrufer fremde Objekte (BOLA)?
- ? Wird die Rollenprüfung durchgesetzt?

Sie beantworten die Laufzeittests von **sectestx** — gegen Ihre Testumgebung, im eigenen Netz.

BELEGT STATT VERMUTET

Jeder Befund nennt die Datei im Export, aus der er stammt. Feststellungen aus der Konfiguration und Hinweise werden getrennt ausgewiesen — nur Feststellungen fließen in die Bewertung. Gelesen wird, was im Gateway tatsächlich greift: Policies zählen nur mit Stage-Zuordnung, Klartext-Backends werden am Routing-Ziel erkannt, mehrere Umgebungen in einem Export bleiben getrennt.

NIS2 · § 30 Abs. 2 Nr. 6 BSIG

NIS2 verlangt, die Wirksamkeit der Sicherheitsmaßnahmen zu bewerten. Für die Konfiguration des Gateways übernimmt wm-audit das automatisch: mit --fail-on bei jeder Änderung in der Pipeline, jeder Bericht mit SHA-256 der geprüften Datei als Nachweis archivierbar. NIS2 insgesamt deckt wm-audit nicht ab.

Bekannte Grenzen. Policies auf Teilbereichen einer API (Scopes) und globale Policies werden nicht unterschieden; Klartext-Routing wird für Straight-Through-Routing bewertet. Getestet mit konstruierten Exporten und öffentlichen Software-AG-Beispielen im Format 10.x, unter macOS und Linux; das Windows-Programm ist unter Wine geprüft. Ein echter Kundenexport steht als Prüfstein noch aus.



wm-audit

POWERED BY
LEANOFY

WEB

leanofy.de/wm-audit

KONTAKT

kontakt@leanofy.de

STAND

Oktober 2026