Sicherheitshinweise - LibreBooking n8n Node

Dieses Dokument erklärt die npm audit Vulnerabilities und wie man damit umgeht.

Inhaltsverzeichnis

Übersicht der Vulnerabilities

Beim Ausführen von `npm audit` werden möglicherweise folgende Vulnerabilities angezeigt:

Critical: form-data

```
form-data  <4.0.1
Severity: critical
Prototype Pollution in form-data
https://github.com/advisories/GHSA-xxx
```

Moderate: lodash

```
lodash  <4.17.21
Severity: moderate
Prototype Pollution in lodash
https://github.com/advisories/GHSA-xxx
```

Warum diese Vulnerabilities existieren

Diese Vulnerabilities kommen **nicht direkt aus diesem Projekt**, sondern sind **transitive Dependencies** von `n8n-workflow` und `n8n-core`.

Dependency-Kette:

```
n8n-nodes-librebooking
  └── n8n-workflow (devDependency für Typen)
        └── axios
              └── form-data (vulnerable version)
        └── lodash (vulnerable version)
```

Wichtig zu verstehen:

1. **n8n-workflow** ist nur als `devDependency` und `peerDependency` deklariert
2. In Produktion verwendet n8n seine **eigene** n8n-workflow Version
3. Die vulnerable Dependencies werden nur beim **Entwickeln** installiert
4. Diese Package werden **nicht** in das finale dist/ Verzeichnis gebündelt

---

Risikoeinschätzung

Für dieses Projekt: NIEDRIGES RISIKO

Aspekt	Risiko	Begründung
Entwicklung	Niedrig	form-data/lodash werden nicht direkt verwendet
Produktion	Sehr niedrig	Keine transtiven Dependencies werden deployed
n8n Runtime	Abhängig von n8n	n8n selbst muss die Vulnerabilities beheben

Warum niedriges Risiko:

1. **form-data Vulnerability:**
   - Betrifft nur das Parsen von multipart/form-data
   - Dieser Node verwendet keine File-Uploads über form-data
   - Die LibreBooking API verwendet JSON für alle Requests

2. **lodash Vulnerability:**
   - Betrifft `_.set()` und `_.setWith()` Funktionen
   - Dieser Node verwendet keine direkten lodash Aufrufe
   - Die Vulnerability erfordert Angreifer-kontrollierten Input

---

Empfehlungen

Für Entwickler:

1. **Warnungen ignorieren** (wenn nicht kritisch):
   `bash`
   `npm install --ignore-scripts`

2. **Audit bei npm install deaktivieren:**
   ```bash
   # Einmalig:
   npm install –no-audit
   ```

Permanent via .npmrc:
echo “audit=false” >> .npmrc
```

1. **Overrides verwenden** (in package.json):
   `json`
   `"overrides": {`
   `"form-data": "^4.0.1",`
   `"lodash": "^4.17.21"`
   `}`

### Für Produktionsumgebungen:

1. **n8n aktuell halten:** Die n8n-Entwickler aktualisieren regelmäßig ihre Dependencies
2. **Nur vertrauenswürdige Inputs:** Keine ungeprüften Daten an die Nodes übergeben
3. **Network Isolation:** n8n Container im isolierten Netzwerk betreiben

---

## Wie man sie beheben kann

### Option 1: Overrides in package.json (empfohlen)

Die package.json enthält bereits Overrides für bekannte Vulnerabilities:

```
"overrides": {
  "form-data": "^4.0.1",
  "lodash": "^4.17.21"
}
```

Option 2: npm audit fix (begrenzt)

```
# Automatische Fixes (nur kompatible Updates)
npm audit fix

# Force Fixes (VORSICHT: kann Breaking Changes einführen)
npm audit fix --force
```

Hinweis: `npm audit fix` kann transitive Dependencies nur begrenzt beheben.

Option 3: Update-Skript verwenden

```
./update-dependencies.sh
```

Das Skript:
- Führt `npm audit fix` aus
- Aktualisiert alle Dependencies
- Testet ob alles funktioniert
- Gibt einen Report

Option 4: Resolutions (für yarn/pnpm)

Wenn Sie yarn statt npm verwenden:
```

```
"resolutions": {
  "form-data": "^4.0.1",
  "lodash": "^4.17.21"
}
```

Produktionsumgebungen

Best Practices:

1. **Docker Image aktuell halten:**
   `bash`
   `docker pull n8nio/n8n:latest`

2. **Regelmäßige Updates:**
   `bash`
   `docker compose pull`
   `docker compose up -d`

3. **Security Scanning:**
   `bash`
   `# Image auf Vulnerabilities prüfen`
   `docker scan n8nio/n8n:latest`
   `# Oder mit Trivy:`
   `trivy image n8nio/n8n:latest`

4. **Netzwerk-Isolation:**
   - n8n nicht direkt im Internet exponieren
   - Reverse Proxy mit TLS verwenden
   - Firewall-Regeln setzen

5. **Zugriffskontrollen:**
   - Starke Passwörter verwenden
   - Basic Auth oder OAuth aktivieren
   - API-Keys für LibreBooking sicher speichern

Sicherheits-Checkliste:

- [ ] n8n Version aktuell?
- [ ] Docker Image aktuell?
- [ ] TLS/HTTPS aktiviert?
- [ ] Starke Passwörter?
- [ ] Netzwerk isoliert?
- [ ] Regelmäßige Backups?

Weiterführende Links

- n8n Security Best Practices (https://docs.n8n.io/hosting/security/)
- npm audit Documentation (https://docs.npmjs.com/cli/v8/commands/npm-audit)

- OWASP Dependency Check (https://owasp.org/www-project-dependency-check/)
- Snyk Vulnerability Database (https://snyk.io/vuln/)

Meldung von Sicherheitsproblemen

Wenn Sie eine Sicherheitslücke **direkt in diesem Projekt** (nicht in Dependencies) finden:

1. **Nicht öffentlich melden** (kein GitHub Issue)
2. Kontaktieren Sie uns direkt per E-Mail
3. Geben Sie Zeit für einen Fix bevor öffentliche Disclosure

Letzte Aktualisierung: Januar 2026