

Mehr Testabdeckung!

über den Tellerrand hinaus

was man eigentlich noch so alles testen?





Christian Kühn

Software - Entwickler

Gute-Laune-Verbreiter

Interessen:
Security, DevOps, Automation



dmTECH GmbH

100% dm-Tochter

komplette Konzern-IT

Hardware

Software

Betrieb



Was sind eigentlich Tests?

„Unter Testen versteht man den Prozess des Planens, der Vorbereitung und der Messung, mit dem Ziel, die Eigenschaften eines IT-Systems festzustellen und den Unterschied zwischen dem tatsächlichen und dem erforderlichen Zustand aufzuzeigen.“

- Pol, Koomen und Spillner,



Tests zeigen Probleme auf

"If we stopped testing right now, we'd have very few cases, if any."

- Donald Trump

*"Program testing can be used to show the presence of bugs,
but never to show their absence!"*

- Edsger W. Dijkstra



Test in Produktion

Test hat Fehler aufgezeigt

Warn-Tag war ein Erfolg!

Verbesserung möglich



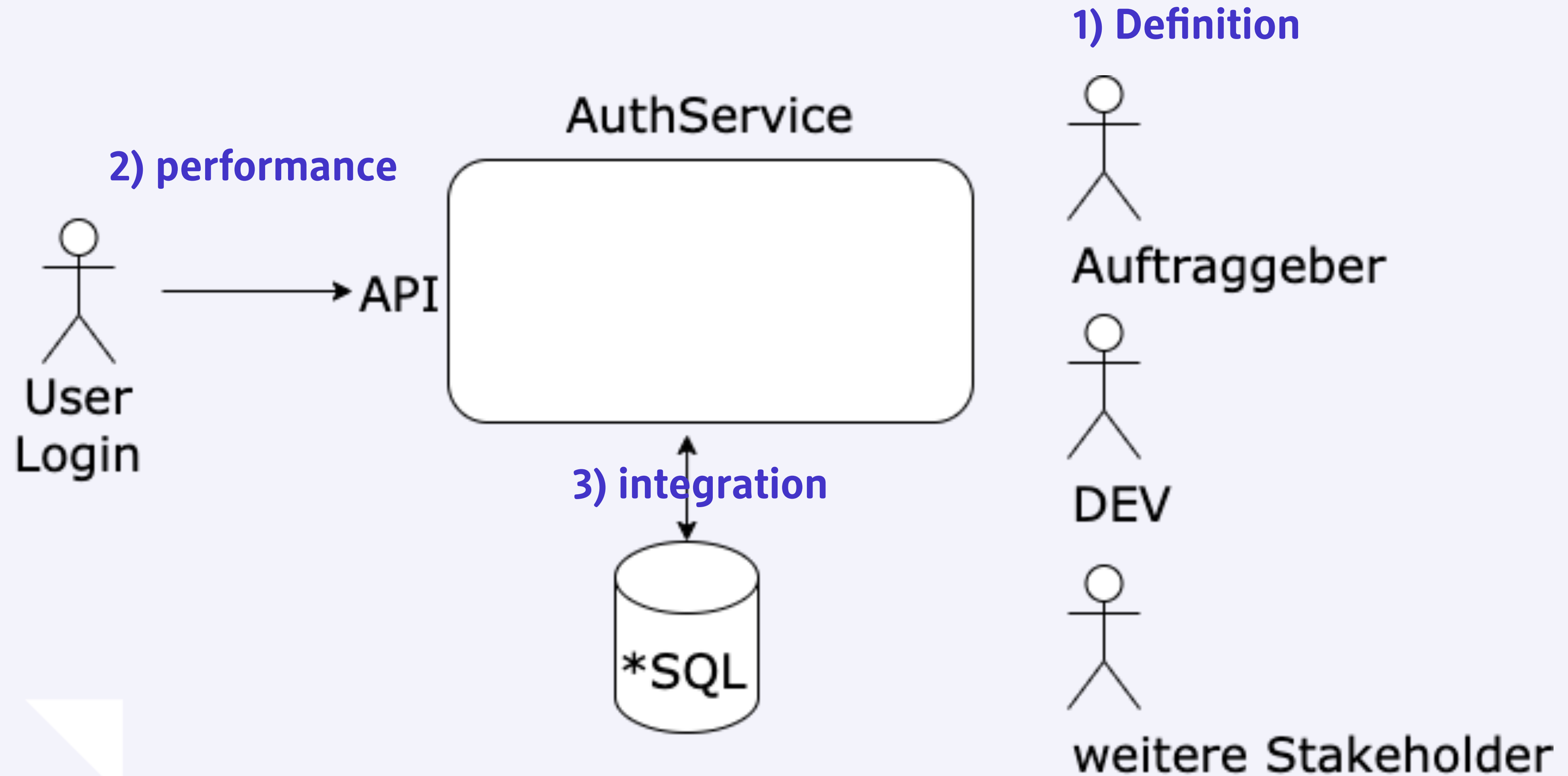
Kennzahlen regelmäßig prüfen

so viele Tests wie nötig

so wenige Tests wie möglich 🤖



Storyline



(* 4) Sandbox anbieten)





BDD

behaviour driven development

**Entwicklungsprozess mit enger
Zusammenarbeit und
Integration aller Stakeholder**

**Fachabteilung definiert Testfälle
mit Dev-Team und Q&A-Team**

**"lesbare" Testbeschreibung
für Nicht-Techniker**

Spezifikation der Software wird gemeinsam ausgearbeitet



Auftraggeber (Fachabteilung, Kunde):
Beschreibung der Business-Prozesse, fachlicher Ansprechpartner

Entwicklungsteam:
Erarbeitung einer Lösungsstrategie und deren Umsetzung

Q&A-Team:
Unterstützung der Lösungsentwicklung, Perspektive Qualität: "What-if?"



Testwerkzeuge

Abbildung der Spezifikation in Form von Given - When - Then

Szenariogrundriss: Ein Kunde versucht sich mit seinem Passwort zu authentifizieren.

Angenommen Ein Kunde startet den Checkout-Prozess

Wenn der Kunde mit der AccountId "12345" sich mit Passwort "hunter2" authentifiziert

Dann antwortet der Auth-Service mit dem Status angemeldet



Szenariogrundriss: Ein Kunde versucht sich mit seinem Passwort zu authentifizieren.

Angenommen Ein Kunde startet den Checkout-Prozess

Wenn der Kunde mit der AccountId "12345" sich mit Passwort "hunter2" authentifiziert

Dann antwortet der Auth-Service mit dem Status angemeldet

```
@Given("Ein Kunde startet den Checkout-Prozess")
public void prepareCheckout() {
    doPreparation();
}

@When("der Kunde mit der AccountId \"{login}\" sich mit Passwort \"{pass}\" authentifiziert")
public void authenticate(String login, String pass) {
    RequestSpecification request = given();
    addAuthorizedAuthentication(request);

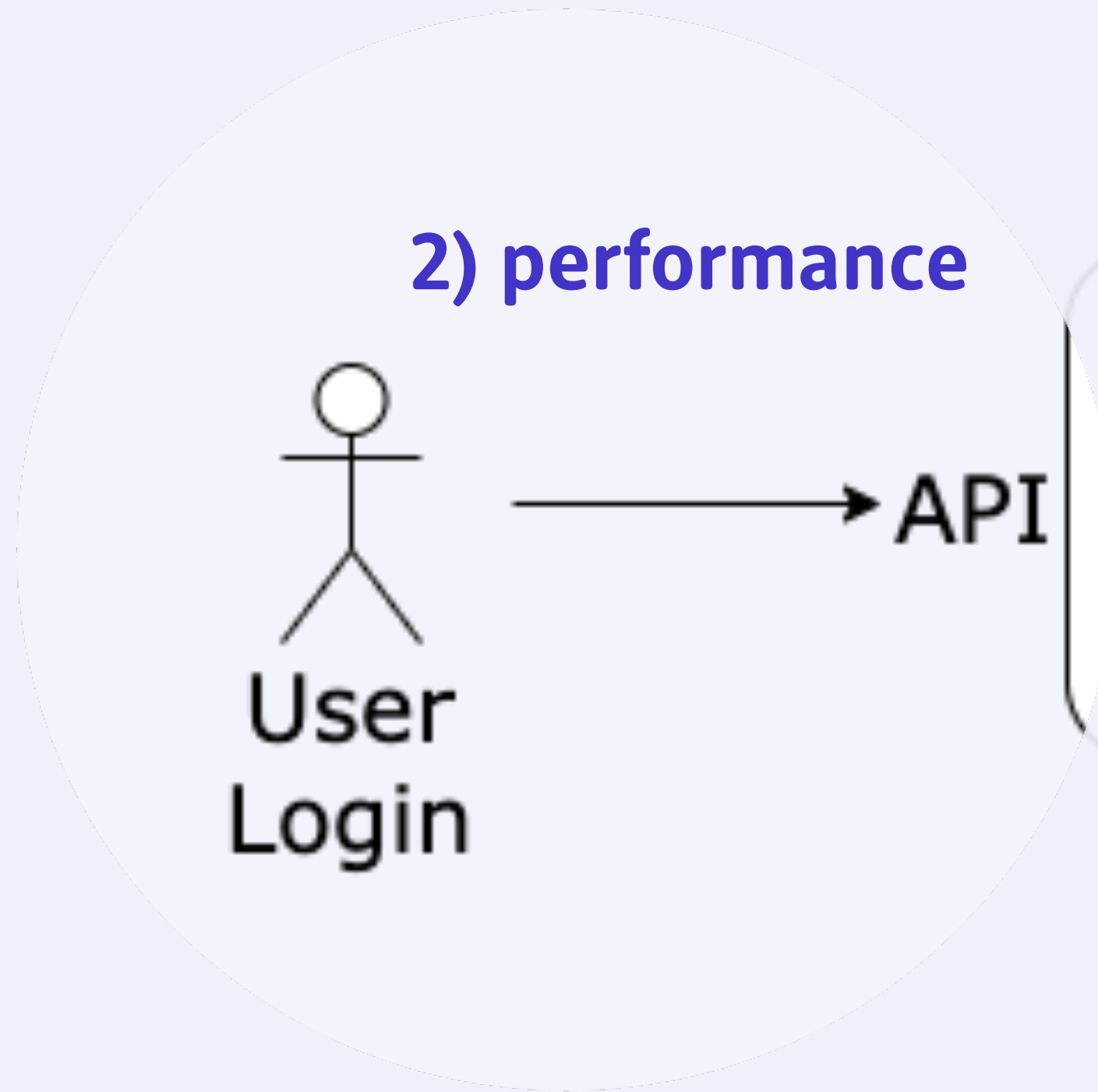
    Map<Object, Object> authenticateRequest = new HashMap<>();
    authenticateRequest.put("login", login);
    authenticateRequest.put("pass", pass);
    request.contentType(ContentType.JSON).body(authenticateRequest);

    responseHolder.setHttpResponse(
        request
            .post(s: "/api/authenticate/")
            .andReturn());

    authentication.setIsAuthenticated(responseHolder.getHttpResponse().jsonPath().get("isAuthenticated"));
}

@Then("antwortet der Auth-Service mit dem Status angemeldet")
public void assertAuthentication() {
    assertThat(authentication.isAuthenticated()).isTrue();
}
```

Storyline



3) integration



1) Definition



Auftraggeber



DEV



weitere Stakeholder

(* 4) Sandbox anbieten)

Gatling

continuous performance testing



in jedem Build

**verschiedene Szenarien
je nach Environment oder Build**

Historie vergleichen



"Load Test as Code"

Scala - basiert (Netty / Akka)

metric-Export*

*** (enterprise-Version oder plugin)**

<https://gatling.io>



Scenario

Beschreibung des Testfalls

```
val successfulAuthentication: ScenarioBuilder = scenario("Successful Authenticate")
    .exec(http("auth")
        .post("/api/authenticate")
        .body(StringBody("""{ "account": "12345", "pass": "hunter2" }"""))).asJson
    .check(status.is(200))
```

```
val failedAuthentication: ScenarioBuilder = scenario("Failed Authenticate")
    .exec(http("auth")
        .post("/api/authenticate")
        .body(StringBody("""{ "account": "12345", "pass": "passw0rd" }"""))).asJson
    .check(status.is(401))
```

Scenario (2)

Kombination mehrerer Szenarien

```
val authenticateCombined: ScenarioBuilder = scenario("Authenticate combo")  
    .randomSwitch(  
        85.0 -> exec(successfulAuthentication),  
        15.0 -> exec(failedAuthentication))
```



Simulation

Konfiguration des Testfalls

- Anzahl der User / Consumer
- httpConfig
- rampUp / -down
- duration



Simulation (Beispiel)

```
val getHttpConfig: () => HttpProtocolBuilder = () => {  
  val httpConfig: HttpProtocolBuilder = http  
    .baseUrl("http://localhost:8080")  
    .userAgentHeader("Gatling")  
    .acceptEncodingHeader("gzip")  
}
```

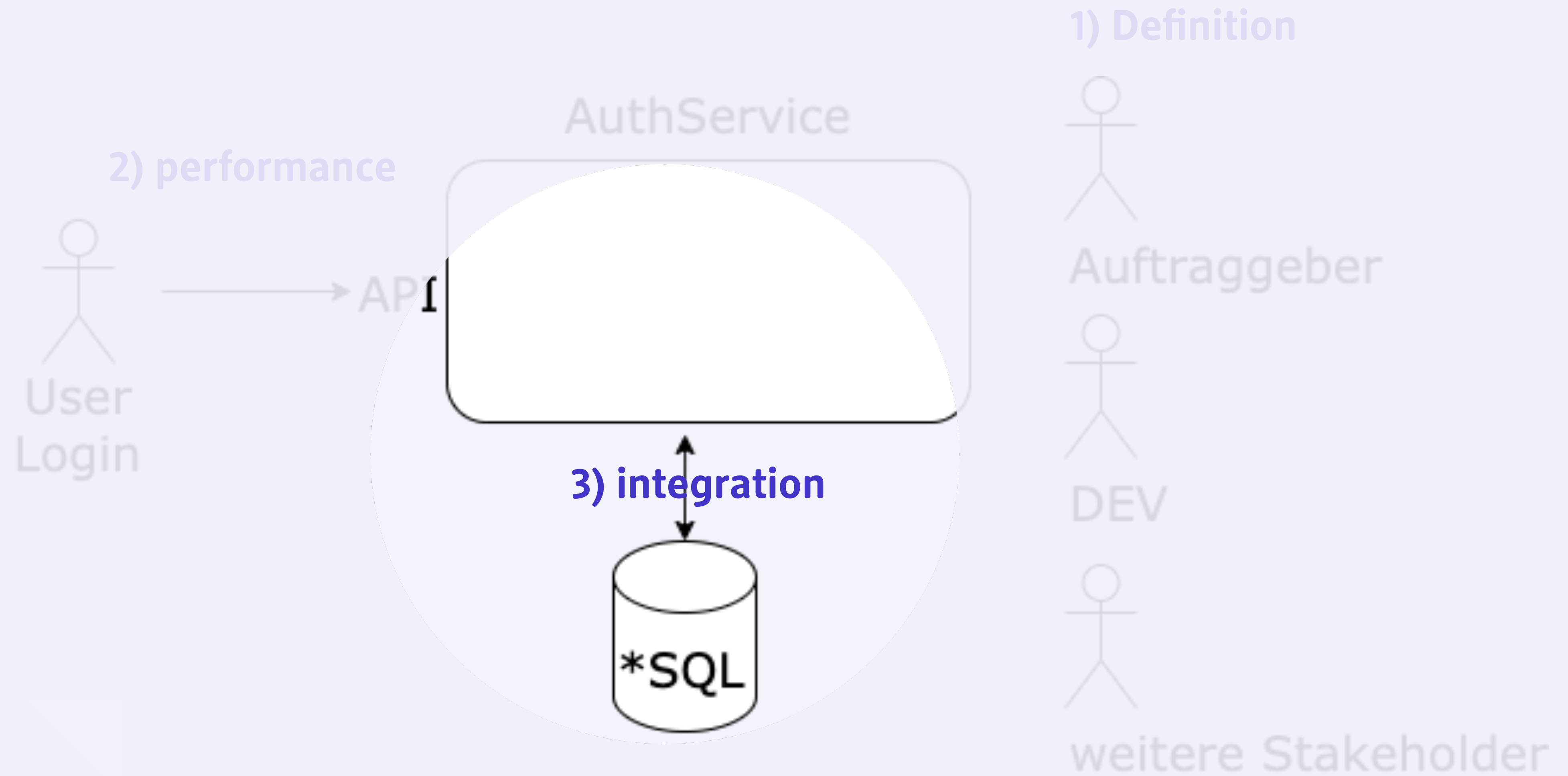


```
setUp(  
  authenticateCombined.inject(  
    rampUsersPerSec(rate1 = 0) to 30 during (10 seconds), // RAMP UP, 3 additional requests per second  
    constantUsersPerSec(rate = 30) during(30 seconds) randomized, // 20 requests per second for 30 seconds  
    rampUsersPerSec(rate1 = 30) to 0 during (10 seconds)) // RAMP Down, 3 less requests per second  
  ).protocols(httpConfig)  
  .assertions(  
    global.failedRequests.percent.lte(threshold = 0.05), // under 5% wrong answers  
    global.responseTime.percentile4.lte(threshold = 1000) // 99% of requests answered quicker than 1000ms  
  )  
)
```

demo



Storyline



(* 4) Sandbox anbieten)



**realitätsnahes Testen auf
Basis von Docker-Containern**

Orchestrierung direkt aus JUnit

**komplexe Testlandschaft möglich
(microservice-Integration)**

Testcontainers
next level Integration testing

Testcontainers



Tests gegen "andere" eigene und fremde Services

**Datenbankoperationen gegen echte
MySQL, MariaDB, PostgreSQL anstatt h2**

**UI-Tests mithilfe containerisierter Browser
Selenium-kompatibel**

<https://www.testcontainers.org/>



viele fertig nutzbare Container im Angebot

z.B. MySQL, MongoDB, Kafka, Elasticsearch, Neo4j, RabbitMQ

Databases ^

[Database containers](#)

- JDBC support
- R2DBC support
- Cassandra Module
- CockroachDB Module
- Couchbase Module
- Clickhouse Module
- DB2 Module
- Dynalite Module
- InfluxDB Module
- MariaDB Module
- MongoDB Module
- MS SQL Server Module
- MySQL Module
- Neo4j Module
- Oracle-XE Module
- OrientDB Module
- Postgres Module
- Presto Module

Docker Compose Module

- Elasticsearch container
- GCloud Module
- Kafka Containers
- Localstack Module
- Mockserver Module
- Nginx Module
- Apache Pulsar Module
- RabbitMQ Module
- Solr Container
- Toxiproxy Module
- Hashicorp Vault Module
- [Webdriver Containers](#)





eigene Container nutzbar

```
private static final GenericContainer authServiceContainer = new GenericContainer(new ImageFromDockerfile()  
    .withFileFromClasspath(path: ".", resourcePath: "/containers/authservice")  
    .withFileFromPath(path: "authservice.jar", Paths.get(System.getProperty("authservice.jar"))))  
    .withEnv("API_KEY", "lala")  
    .withEnv("spring.profiles.active", "testcontainers")  
    .withEnv("SSL_ENABLED", "false")  
    .withEnv("DB_PROVIDER", "mysql"))
```

```
new GenericContainer(DockerImageName.parse("jboss/wildfly:9.0.1.Final"))
```





Verwendung

Verwaltung über integrierten docker-Client

eigene Container-Instanzen pro Test oder "shared"

Container manuell start/stopbar

`container.start()`

Containerklassen sind `AutoCloseable`





demo

Danke Sandra!

<https://blog.sandra-parsick.de/2020/05/21/using-testcontainers-in-spring-boot-tests-for-database-integration-tests/>

Storyline

1) Definition



Auftraggeber



DEV



weitere Stakeholder

2) performance



User
Login

→ API

AuthService



3) integration



(* 4) Sandbox anbieten)

Wiremock

API - Simulator (HTTP)

**vorbereitete Responses
abhängig vom Request**

record / playback

JUnit oder Standalone



Http-Server bzw. -Services mocken

**Responses abhängig vom Request
z.B. bestimmtes Mapping auf auth-header**

Konfiguration in Java oder JSON



```
{
  "request": {
    "url": "/api/authenticate",
    "method": "POST",
    "bodyPatterns": [
      {
        "matchesJsonPath": "$[?(@.pass == 'hunter2')]"
      }
    ]
  },
  "response": {
    "status": 200,
    "body": "{ \"isAuthenticated\": \"true\" }",
    "fixedDelayMilliseconds": 350,
    "headers": {
      "Content-Type": "application/json;charset=UTF-8",
      "Cache-Control": "no-cache, no-store, max-age=0, must-revalidate",
      "Pragma": "no-cache",
      "Expires": "0",
      "Date": "{now timezone='GMT' format='EEE, d MMM yyyy HH:mm:ss z'}"
    }
  },
  "metadata": {
    "description": "Nutzer mit richtigem Passwort logt sich ein."
  }
}
```



Consumer testen für Drittanbieter-Services

Fehler und Exceptions in Request einbauen

Latenzen simulieren (z.B. für performance-Tests)

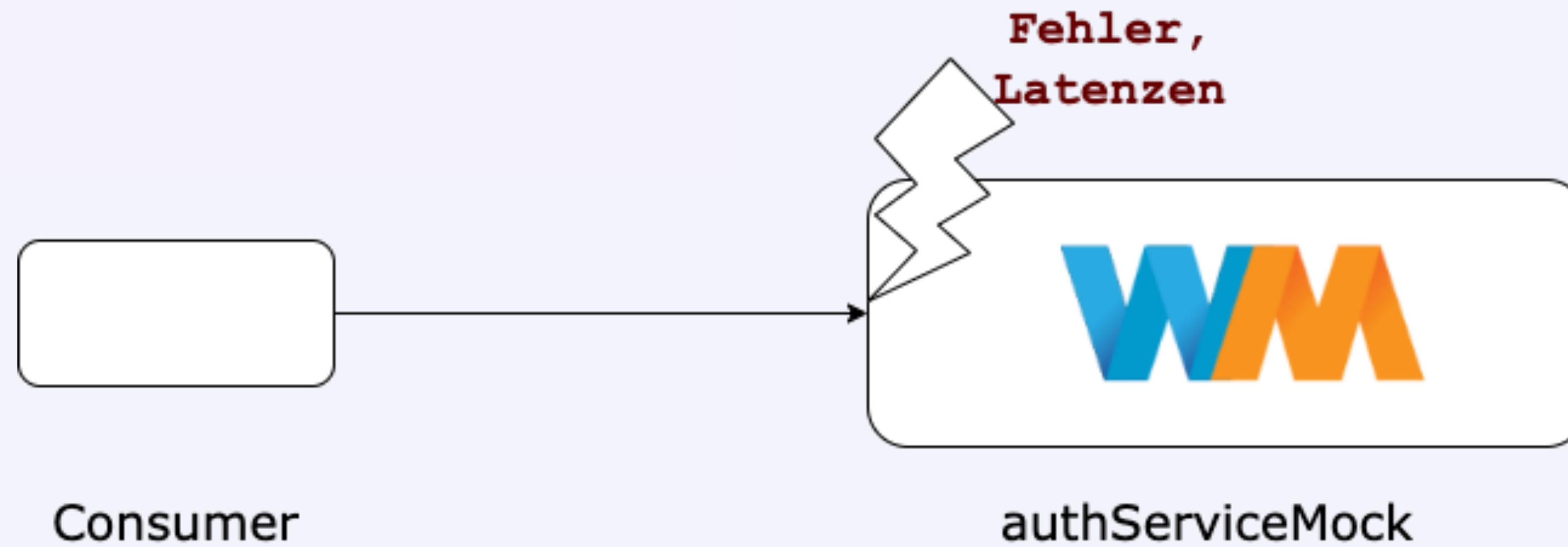


verschiedene Zustände pro Scenario möglich

**Steuerung und Verifizieren
über "Admin"-API**



Sandbox für den eigenen Service anbieten



Bonus!

Security "gratis"



DevSecOps*

**bekannte Schwachstellen
in Abhängigkeiten finden**

**Anwendung absichern
ohne Security-Kenntnisse**

dependency-check
software supply chain absichern

OWASP - Projekt

**Analyse von externen Abhängigkeiten
auf bekannte Schwachstellen**





```
[INFO] Analysis Started
[INFO] Finished Archive Analyzer (0 seconds)
[INFO] Finished File Name Analyzer (0 seconds)
[INFO] Finished Jar Analyzer (0 seconds)
[INFO] Finished Dependency Merging Analyzer (0 seconds)
[INFO] Finished Version Filter Analyzer (0 seconds)
[INFO] Finished Hint Analyzer (0 seconds)
[INFO] Created CPE Index (2 seconds)
[INFO] Finished CPE Analyzer (3 seconds)
[INFO] Finished False Positive Analyzer (0 seconds)
[INFO] Finished NVD CVE Analyzer (0 seconds)
[INFO] Finished Sonatype OSS Index Analyzer (0 seconds)
[INFO] Finished Vulnerability Suppression Analyzer (0 seconds)
[INFO] Finished Dependency Bundling Analyzer (0 seconds)
[INFO] Analysis Complete (5 seconds)
```

[WARNING]

One or more dependencies were identified with known vulnerabilities in hello:

```
jackson-databind-2.9.6.jar (pkg:maven/com.fasterxml.jackson.core/jackson-databind@2.9.6, cpe:2.3:a:fasterxml:jackson:2.9.6:*:*:*:*:*:*:**, cpe:2.3:a:fasterxml:jack
18-14721, CVE-2018-19360, CVE-2018-19361, CVE-2018-19362, CVE-2019-12086, CVE-2019-12384, CVE-2019-12814, CVE-2019-14379, CVE-2019-14439, CVE-2019-14540, CVE-2019
2019-20330, CVE-2020-10672, CVE-2020-10673, CVE-2020-10968, CVE-2020-10969, CVE-2020-11111, CVE-2020-11112, CVE-2020-11113, CVE-2020-11619, CVE-2020-11620, CVE-20
-2020-9546, CVE-2020-9547, CVE-2020-9548
log4j-api-2.10.0.jar (pkg:maven/org.apache.logging.log4j/log4j-api@2.10.0, cpe:2.3:a:apache:log4j:2.10.0:*:*:*:*:*:**) : CVE-2020-9488
snakeyaml-1.19.jar (pkg:maven/org.yaml/snakeyaml@1.19, cpe:2.3:a:snakeyaml_project:snakeyaml:1.19:*:*:*:*:*:**) : CVE-2017-18640
spring-core-5.0.8.RELEASE.jar (pkg:maven/org.springframework/spring-core@5.0.8.RELEASE, cpe:2.3:a:pivotal_software:spring_framework:5.0.8:release:*:*:*:*:**, cpe
rk:5.0.8:release:*:*:*:*:**) : CVE-2018-15756, CVE-2020-5398, CVE-2020-5421
spring-security-core-5.0.7.RELEASE.jar (pkg:maven/org.springframework.security/spring-security-core@5.0.7.RELEASE, cpe:2.3:a:pivotal_software:spring_security:5.0.
tomcat-embed-core-8.5.32.jar (pkg:maven/org.apache.tomcat.embed/tomcat-embed-core@8.5.32, cpe:2.3:a:apache:tomcat:8.5.32:*:*:*:*:*:**, cpe:2.3:a:apache_software_
18-11784, CVE-2019-0199, CVE-2019-0221, CVE-2019-0232, CVE-2019-10072, CVE-2019-12418, CVE-2019-17563, CVE-2020-11996, CVE-2020-13934, CVE-2020-13935, CVE-2020-13
```

See the dependency-check report for more details.

```
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 10.747 s
[INFO] Finished at: 2020-11-22T01:56:12+01:00
[INFO] -----
```

Published Vulnerabilities[CVE-2018-1000873](#)

Fasterxml Jackson version Before 2.9.8 contains a CWE-20: Improper Input Validation vulnerability in Jackson-Modules-Java8 that can result in Causes a denial-of-service (DoS). This attack appear to be exploitable via The victim deserializes malicious vulnerability appears to have been fixed in 2.9.8.

CWE-20 Improper Input Validation

CVSSv2:

- Base Score: MEDIUM (4.3)
- Vector: /AV:N/AC:M/Au:N/C:N/I:N/A:P

CVSSv3:

- Base Score: MEDIUM (6.5)
- Vector: /AV:N/AC:L/PR:N/UI:R/S:U/C:N/I:N/A:H

References:

- CONFIRM - https://bugzilla.redhat.com/show_bug.cgi?id=1665601
- CONFIRM - <https://security.netapp.com/advisory/ntap-20200904-0004/>
- MISC - <https://github.com/FasterXML/jackson-modules-java8/issues/90>
- MISC - <https://github.com/FasterXML/jackson-modules-java8/pull/87>
- MISC - <https://www.oracle.com/security-alerts/cpuoct2020.html>
- MISC - <https://www.oracle.com/technetwork/security-advisory/cpujul2019-5072835.html>
- MISC - <https://www.oracle.com/technetwork/security-advisory/cpuoct2019-5072832.html>
- MLIST - [\[drill-dev\] 20191017 Dependencies used by Drill contain known vulnerabilities](#)
- MLIST - [\[drill-dev\] 20191021 \[jira\].\[Created\].\(DRILL-7416\) Updates required to dependencies to resolve potential security vulnerabilities](#)
- MLIST - [\[drill-issues\] 20191021 \[jira\].\[Created\].\(DRILL-7416\) Updates required to dependencies to resolve potential security vulnerabilities](#)
- MLIST - [\[nifi-commits\] 20191113 svn commit: r1869773 - /nifi/site/trunk/security.html](#)
- MLIST - [\[nifi-commits\] 20200123 svn commit: r1873083 - /nifi/site/trunk/security.html](#)
- MLIST - [\[pulsar-commits\] 20190416 \[GitHub\] \[pulsar\] one70six opened a new issue #4057: Security Vulnerabilities - Black Duck Scan - Pulsar v.2.3.1](#)
- N/A - [N/A](#)
- OSSINDEX - [\[CVE-2018-1000873\] Improper Input Validation](#)

Vulnerable Software & Versions: ([show all](#))

- [cpe:2.3:a:fasterxml:jackson-databind:*:*:*:*:* versions up to \(excluding\) 2.9.8](#)
- ...

[CVE-2018-14718](#)

FasterXML jackson-databind 2.x before 2.9.7 might allow remote attackers to execute arbitrary code by leveraging failure to block the slf4j-ext class from polymorphic deserialization.

CWE-502 Deserialization of Untrusted Data

CVSSv2:

- Base Score: HIGH (7.5)
- Vector: /AV:N/AC:L/Au:N/C:P/I:P/A:P

CVSSv3:

- Base Score: CRITICAL (9.8)
- Vector: /AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

References:

- BID - [106601](#)
- BUGTRAQ - [20190527 \[SECURITY\]\[DSA 4452-1\] jackson-databind security update](#)
- CONFIRM - <https://github.com/FasterXML/jackson-databind/commit/87d29af25e82a249ea15858e2d4ecbf64091db44>
- CONFIRM - <https://github.com/FasterXML/jackson-databind/issues/2097>
- CONFIRM - <https://github.com/FasterXML/jackson/wiki/Jackson-Release-2.9.7>
- CONFIRM - <https://security.netapp.com/advisory/ntap-20190530-0003/>
- CONFIRM - <https://www.oracle.com/technetwork/security-advisory/cpujan2019-5072801.html>
- DEBIAN - [DSA-4452](#)
- MISC - <https://www.oracle.com/security-alerts/cpujan2020.html>



Danke für euer Interesse! :)

Fragen?

Freue mich über Feedback chris@clowncomputing.de  [@CYxChris](https://twitter.com/CYxChris)