



Sicherer Anschlussknoten des Data Consumers Integrationsanleitung Bundesverwaltungsamt

Version 1.0.1, Datum 13.11.2025

Inhaltsverzeichnis

Änderungshistorie	1
1. Einleitung	1
2. Überblick	2
2.1. Fachliche Beschreibung	2
2.2. Technische Beschreibung	2
3. Installation der Anwendung	4
3.1. Installation	4
4. Regulärer Wirkbetrieb	8
4.1. Logging	8
4.2. Statusprüfung der Anwendungen	8
4.3. Monitoring	8
4.4. Backup und Recovery	9
4.5. Hinweise für Störungsdiagnose und -behandlung	9
Anhang	10
Anhang A: Technisches Glossar	10
Anhang B: Fachliches Glossar	10
Anhang C: Betrieblich-Relevante Konfigurationsparameter	11
Quellen	15

Änderungshistorie

Version	Änderungen
0.1	• Initiale Erstellung des Dokuments
0.2	• Einarbeitung Review Kommentare
0.9	• Dokument finalisiert
1.0	• Dokument abgenommen
1.0.1	• Anpassungen in der Beispiel-Konfigurationsdatei • Hinzufügen von Konfigurationsparametern für das Rate-Limiting der Anwendung • Anpassung von neuen Links und Referenzen auf OpenCode

1. Einleitung

Das vorliegende Dokument enthält die Integrationsanleitung für die Anwendung Sicherer Anschlussknoten des Data Consumers in der jeweils aktuell veröffentlichten Version. Es fungiert als Hilfestellung für den Betrieb der Anwendung.

Das Kapitel [Überblick](#) bietet einen groben fachlichen, technischen sowie betrieblichen Überblick über die Anwendung und liefert Verweise auf weitergehende Dokumente.

Das Kapitel [Installation der Anwendung](#) beschreibt die Installation und Konfiguration der Anwendung in der Laufzeitumgebung.

Relevant für Betrieb und Softwareentwicklung

Das Kapitel [Regulärer Wirkbetrieb](#) liefert Informationen für den Betrieb der Anwendung und gibt Hilfestellung bei der Fehleranalyse

Relevant für Betrieb und Softwareentwicklung

Der [Anhang](#) enthält ein fachliches und technisches Glossar.

2. Überblick

In diesem Kapitel wird aus fachlicher und technischer Sicht ein Überblick über die Geschäftsprozesse gegeben, die mithilfe des SAK-DCs durchgeführt werden.

2.1. Fachliche Beschreibung

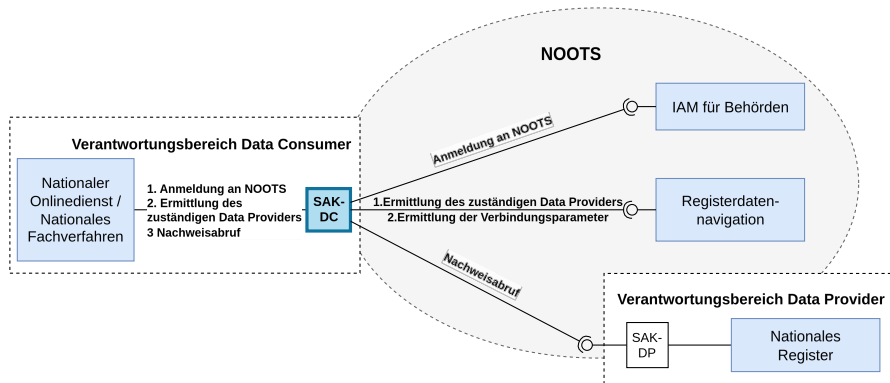


Abbildung 1. SAK-DC: Fachlicher Kontext

Die fachlichen Aufgaben des SAK-DCs sind die Durchführung des Nachweisabrufs unter Einhaltung des obligatorischen Sicherheitsprotokolls.

Im Einzelnen:

- der Abruf des NOOTS-Zugriffstokens vom IAM für Behörden
- die Abfrage des Data Providers und der benötigten Verbindungsparameter
- der eigentliche Nachweisabruf.

2.2. Technische Beschreibung

Der SAK-DC ermöglicht via das Anschlussprotokoll (bestehend aus REST-APIs) den sicheren Zugang zum NOOTS-Netzwerk für einen Data Consumer.

- Zur Absicherung der Zugriffe auf SAK-DC kommen **Basic Auth** und **OAuth 2.0** zum Einsatz
 - Bei einer gleichzeitigen Nutzung von **Basic Auth** und **OAuth 2.0** (Zugriffstoken) auf den Schnittstellen `/evidence-offer` und `/evidence` müssen beide Header-Werte in **einem** HTTP-Header angegeben werden:

Beispiel 1. Nutzung von **Bearer**- und **Basic**-Schemas in einem Header

```
Authorization: Bearer [[Zugriffstoken]],Basic
```

[[Basic-Auth]]

- Zur Absicherung der Netzwerk-Kommunikation zwischen DC und SAK-DC kommt **(m)TLS** zum Einsatz

In der folgenden Matrix sind die Sicherheitsmechanismen in einer Matrix aufgeschlüsselt. Hier kann eingesehen werden, welcher Sicherheitsmechanismus wann eingesetzt werden sollte oder muss.

Tabelle 1. Aktivierung / Erzwingung der Sicherheitsmechanismen je Endpunkt

Endpunkt	mTLS	OAuth 2.0	Basic Auth
/token	optional (empfohlen)	-	optional
/de/evidence-offer	optional (empfohlen)	erforderlich	optional
/de/evidence	optional (empfohlen)	erforderlich	optional

Die hierzu benötigten Zugangsdaten und Zertifikate müssen bei der Installation bereitgestellt und konfiguriert werden. Die benötigten Zugriffstoken für die OAuth 2.0 Authentifizierung müssen zur Laufzeit über die **/token**-Schnittstelle abgerufen werden.

HINWEIS

- Die Übertragung der Nachweisdaten erfolgt ausschließlich über den geschützten Weg zwischen SAK-DC und SAK-DP
- Der Anbindung an den EU-Kontext ist derzeit nicht Teil der Anwendung

Weitere technische Details können im [Anschlusskonzept Data Consumer \(AD-NOOTS-01\)](#) nachgelesen werden.

3. Installation der Anwendung

Dieses Kapitel beschreibt die Installation der Anwendung.

3.1. Installation

Die nachfolgende Installationsanleitung beschreibt die Erstinstallation auf einem Linux-Server (keine Benutzeroberfläche) mit Zugriff mittels **ssh**.

WICHTIG

Halten Sie sich hier an Ihre betrieblichen Vorgaben für einen sicheren Betrieb. Die hier beschriebenen Schritte dienen nur als Beispiel.

Der SAK-DC wird als ausführbare JAR-Datei (Java Archive) bereitgestellt.

Die JAR-Datei sollte in der Laufzeitumgebung abgelegt werden, in unserem Beispiel ist das **/opt/sak-dc**

Kopieren Sie die JAR-Datei mit der entsprechenden Version auf den Zielrechner, beispielsweise mit **scp**:

```
scp sak-dc-VERSION-TAG.jar admin@laufzeitumgebungshost:~
```

Loggen Sie sich per **ssh** auf dem Rechner ein:

```
ssh admin@laufzeitumgebungshost:~
```

Die Berechtigung sollte nur auf das Nötigste beschränkt werden, damit nur der ausführende User der Serveranwendung Leserechte hat. In unserem Beispiel läuft die Serveranwendung mit dem User **sak-dc** der der Linux Gruppe **sak-dc** zugehört.

```
sudo mkdir /opt/sak-dc
sudo chown root.sak-dc /opt/sak-dc
sudo chmod 750 /opt/sak-dc

sudo mv ~/sak-dc-VERSION-TAG.jar /opt/sak-dc

sudo chown root.sak-dc /opt/sak-dc/sak-dc-VERSION-
TAG.jar
sudo chmod 640 /opt/sak-dc/sak-dc-VERSION-TAG.jar
```

WICHTIG

Die Anwendung sollte unter keinen Umständen über den **root**-User des Systems, sondern idealerweise

mittels eines eigenen Benutzers mit möglichst minimalen Rechten ausgeführt werden.

3.1.1. Systemvoraussetzungen

Die Systemvoraussetzungen für den Betrieb sind abhängig von der Anzahl der gleichzeitigen Zugriffe auf den SAK-DC. Je nach Mengengerüst können diese erheblich von den Mindestanforderungen abweichen.

Tabelle 2. Systemvoraussetzungen

Name	Mindestanforderungen
Java-JRE	JRE 21, getestet mit Adoptium Temurin
Betriebssystem	Linux amd64
Arbeitsspeicher	1024MB
Speicherplatz	512MB
Prozessor	2 GHz Dual Core, 64-Bit
Netzwerk	Internetverbindung (direkt oder über Proxy) auf Port 443

3.1.2. Start und Konfiguration der Anwendung

Die Konfigurationen des SAK-DC findet mittels einer bereitgestellten **application.properties** statt, welche auf die Laufzeitumgebung angepasst werden muss. Die Parameter sind in der Datei beschrieben und enthalten Beispielwerte, an denen sich orientiert werden kann. Die Konfigurationsdatei muss als Parameter übergeben werden. Die Dateirechte auf die Konfigurationsdatei sollten auch hier so gesetzt werden, sodass nur die Anwendung darauf zugreifen kann.

```
java -jar sak-dc-VERSION-TAG.jar
--spring.config.additional
-location=application.properties
```

Weiterführende Hinweise zur Konfiguration

Im oben genannten Beispiel wird davon ausgegangen, dass die **application.properties**-Datei im identischen Verzeichnis wie die Anwendung (.jar) selbst liegt. Wenn die Konfiguration aus einem anderen Verzeichnis geladen werden soll, kann dieses sowohl relational, sowie absolut angegeben werden. Außerdem können mehrere Dateien, auch *optional* angegeben werden. Hier muss die

Reihenfolge beachtet werden. Dateien, welche früher in der Kette angegeben werden, werden durch spätere Dateien überschrieben (je nach Konfigurationsschlüssel)

```
java -jar sak-dc-VERSION-TAG.jar
--spring.config.additional-location=\

    #Konfigurations-Datei im gleichen Ordner; muss
    vorhanden sein
    application.properties,\

    #externe Konfigurations-Datei ; muss vorhanden
    sein
    /opt/default.properties,\

    #externe Konfigurations-Datei ; optional,
    produziert keinen Fehler, wenn nicht vorhanden
    optional:/opt/optional.properties
```

Weitere Informationen zu der Angabe von externen Konfigurationen können der offiziellen [Spring Boot Dokumentation](#); Kapitel "Externalized Configuration"; Abschnitt "External Application Properties" entnommen werden.

Die Anwendung sollte nun gestartet sein und folgende Meldung in der Ausgabe erscheinen:

Komponente SAK-DC erfolgreich gestartet

3.1.3. Proxykonfiguration

Wird für die Kommunikation nach Außen ein Proxyserver benötigt, muss dieser beim Start der Anwendung konfiguriert werden, nicht in der **application.properties**.

```
java -Dhttps.proxyHost=IP/HOSTPROXY
-Dhttps.proxyPort=PROXYPORT
-Dhttp.nonProxyHosts=PROXYAUSNAHMEN -jar sak-dc-VERSION-
TAG.jar --spring.config.additional
-location=application.properties
```

3.1.4. Endpunkte

In der Standardkonfiguration ist die Serveranwendung über Port **8080** sowie der Managementendpunkt über Port **9000** erreichbar.

WICHTIG

Diese Netzwerkports sollten über geeignete Firewallregeln abgesichert werden, um unbefugten Zugriff zu unterbinden. Halten Sie sich auch hier an Ihre betrieblichen Vorgaben für einen sicheren Betrieb.

In den offiziellen Test- und Produktionsumgebungen von NOOTS ist es verpflichtend, die Endpunkte nicht ungeschützt und unverschlüsselt zu betreiben. Es können hierbei verschiedenen Varianten genutzt werden. Je nach vorhandener Infrastruktur kann auch ein Betrieb hinter einem Proxy sinnvoll sein. In einer lokalen Teststellung ohne Anbindung gegen das NOOTS-System (z.B. mittels Mocks) können die Endpunkte auch weniger abgesichert betrieben werden.

Tabelle 3. Absicherung der Endpunkte

Art	Erklärung
HTTPS	Die Konfiguration erfolgt über die <code>application.properties</code>
mTLS	Die Konfiguration erfolgt über die <code>application.properties</code>
HTTP Basic-Auth	Die Konfiguration erfolgt über die <code>application.properties</code>
Die obigen Methoden können bei Bedarf durch Apache-, Nginx oder einen anderen geeigneten Reverse Proxy abgebildet werden	Die Konfiguration erfolgt über das externe genutzte System

3.1.5. Start und Stop

Damit die Anwendung nicht immer manuell über die Konsole gestartet werden muss, können Mittel wie zum Beispiel `systemd` oder `init.d` genutzt werden.

Wir empfehlen hierzu die offizielle [Spring Boot Dokumentation; Kapitel "Installing Spring Boot Applications"](#).

4. Regulärer Wirkbetrieb

4.1. Logging

Der SAK-DC verwendet standardmäßig die Konsole für Logausgaben. Das Log-Nachrichtenformat entspricht dem [Elastic Common Schema \(ECS\)](#).

4.2. Statusprüfung der Anwendungen

Der SAK-DC stellt die folgenden Endpunkte über den Management Port zur Statusprüfung bereit:

HINWEIS

Diese Endpunkte sollten nur intern (z.B. hinter einer Firewall) betrieben werden. Eine Freigabe für einen öffentlichen, nicht regulierten Zugriff ist zu vermeiden!

Tabelle 4. Status-Endpunkte

Endpunkt	Beschreibung
<code>/actuator/health</code>	Gesamtstatus der Anwendung. Liefert <code>{"status":"UP DOWN"}</code> als Antwort

4.3. Monitoring

Der SAK-DC kann zusätzliche Metriken liefern. Es stehen folgende Endpunkte zur Verfügung:

- `/metrics`
- `/prometheus` ([Spring Boot Dokumentation](#); Kapitel "Prometheus (prometheus)")

Die Endpunkte können über die `application.properties` aktiviert werden und sind über den Management-Port erreichbar.

Es werden folgende Standard-Metriken bereitgestellt:

HINWEIS

Diese Endpunkte sollten nur intern (z.B. hinter einer Firewall) betrieben werden. Eine Freigabe für einen öffentlichen, nicht regulierten Zugriff ist zu vermeiden!

Tabelle 5. Standard-Metriken

Metriken	Beschreibung	Path
JVM	Metriken zum Speicherverbrauch, zur Garbage Collection, Threads und geladenen Klassen.	<code>/actuator/metrics/jvm/*</code>
System	Metriken zur CPU-Auslastung, dem Dateisystem und der Uptime der Anwendung.	<code>/actuator/metrics/disk/ /actuator/metrics/system/ /actuator/metrics/process/*</code>
Application Startup	Metriken zur Startzeit der Anwendung.	<code>/actuator/metrics/application/*</code>
Logger	Metriken zu Logging-Events.	<code>/actuator/metrics/logback/events</code>
REST Services	Metriken zu aufgerufenen REST-Services.	<code>/actuator/metrics/http/*</code>

4.4. Backup und Recovery

Wir empfehlen mindestens folgende Daten zu sichern:

- Angepasste `application.properties` die für den Betrieb verwendet wird.
- Logs der Anwendung unter Beachtung der geltenden Datenschutzbestimmungen.

4.5. Hinweise für Störungsdiagnose und -behandlung

4.5.1. Debug-Logging

Die Konfiguration wird mit Log-Level **INFO** ausgeliefert, d.h. das Loggen von Debug-Ausgaben ist ausgeschaltet. Es kann aber mittels der `application.properties logging.level.root=DEBUG` eingeschaltet werden.

ACHTUNG

Das Logging sollte auf der Produktivumgebung nie auf **DEBUG** gestellt werden. Aufgrund der umfangreichen Datenausgabe könnte es zu Performance-Problemen kommen. Gleichzeitig ist nicht sichergestellt, dass keine sicherheitsrelevanten Daten ausgegeben werden.

Anhang

Anhang A: Technisches Glossar

HINWEIS

Das hier hinterlegte Glossar ist übergreifend für alle Dokumente des SAKs identisch

Tabelle 6. Technisches Glossar SAK-DC

Begriff	Bedeutung
Laufzeitumgebung	Umgebung, in der ein Service oder eine Anwendung betrieben werden kann.

Anhang B: Fachliches Glossar

HINWEIS

Das hier hinterlegte Glossar ist übergreifend für alle Dokumente des SAKs identisch

Tabelle 7. Fachliches Glossar SAK-DC

Begriff	Bedeutung
Data Consumer	Ein Data Consumer ist ein NOOTS-Teilnehmer, welcher Nachweise aus dem System, konkret den Data Providern, abrufen möchte.
Data Provider	Ein Data Provider ist ein NOOTS-Teilnehmer der den Data Consumern Nachweise zur Verfügung stellt.
Identity Access Management für Behörden (IAM-B)	Das IAM für Behörden ist eine zentrale Komponente des NOOTS. Sie stellt für den Zugriff auf andere NOOTS-Komponenten und Data Provider vertrauenswürdige und zuverlässige Zugriffstoken für Data Consumer aus, die den Zugriff des DC auf die NOOTS-Komponenten und die DPs autorisiert, die Identität des DCs bestätigt und die Berechtigungen des DCs gegenüber den NOOTS-Komponenten und DPs beinhaltet.
Registerdatennavigation (RDN)	Die Registerdatennavigation ist eine zentrale Komponente des NOOTS. Sie liefert auf Anfrage die Information, von welchem technischen Dienst einer Behörde ein gesuchter Nachweistyp abgerufen werden kann. Dazu übermittelt die RDN notwendige Verbindungsparameter an den anfragenden Data Consumer.
Nachweis	Nachweise sind Unterlagen und Daten in elektronischer Form, die zur Ermittlung des Sachverhaltes in Verwaltungsverfahren geeignet sind.

Begriff	Bedeutung
National-Once-Only-Technical-System (NOOTS)	Das NOOTS ist ein gemeinsames informationstechnisches System aus IT-Komponenten, Schnittstellen und Standards, das öffentlichen Stellen den Abruf und die Übermittlung von elektronischen Nachweisen und Daten national und grenzüberschreitend aus Datenbeständen öffentlicher Stellen, zur Erfüllung öffentlicher Aufgaben ermöglicht.
Sicherer Anschlussknoten (SAK)	Ein sicherer Anschlussknoten (SAK) ist eine für Data Consumer - und Provider bereitgestellte NOOTS-Komponente, die dezentral im Verantwortungsbereich der NOOTS-Teilnehmer betrieben wird. Er ermöglicht den NOOTS-Teilnehmern über ein interoperables Anschlussprotokoll einen einfachen und sicheren Zugriff auf die NOOTS Transportinfrastruktur.

Anhang C: Betrieblich-Relevante Konfigurationsparameter

```
# -----
# SAK-DC Transportprotokoll Einstellungen
# -----
# Zugewiesene NOOTS-Komponenten-ID des Sicheren
Anschlussknotens
sakdc.id =

# SAK-Client TLS-Auth
# Hardware-basierte Schlüsselspeicher (empfohlen)
spring.ssl.bundle.jks.client.keystore.type = PKCS11
spring.ssl.bundle.jks.client.keystore.password =
spring.ssl.bundle.jks.client.key.alias = sakdc-client-
crt
# Datei-basierte Schlüsselspeicher (nicht empfohlen,
bitte nur PKCS12 in diesem Fall verwenden)
#spring.ssl.bundle.jks.client.keystore.type = PKCS12
#spring.ssl.bundle.jks.client.keystore.location = sakdc-
client-cert.p12
#spring.ssl.bundle.jks.client.keystore.password =

# Truststore mit Zertifikaten, die der SAK bei
ausgehenden Aufrufen vertrauen darf.
# Optional; das System-Truststore des JRE wird
verwendet, falls nicht gesetzt.
#spring.ssl.bundle.jks.client.truststore.type = PKCS12
#spring.ssl.bundle.jks.client.truststore.location =
sakdc-client-truststore.p12
```

```
#spring.ssl.bundle.jks.client.truststore.password =

# Truststore-Konfiguration für die NOOTS-Zugriffstoken-
Validierung
spring.ssl.bundle.jks.iamsigncert.truststore.type =
PKCS12
spring.ssl.bundle.jks.iamsigncert.truststore.location =
iam-sign-cert.p12
spring.ssl.bundle.jks.iamsigncert.truststore.password =
#spring.ssl.bundle.pem.iamsigncert.truststore.certificat
e = iam-sign-cert.crt
#spring.ssl.bundle.pem.iamsigncert.truststore.certificat
e = -----BEGIN CERTIFICATE-----....

# Truststore-Konfiguration für die NOOTS-Zuständigkeit-
/Verbindungstoken (RDN-Token) Validierung
spring.ssl.bundle.jks.rdnsigncert.truststore.type =
PKCS12
spring.ssl.bundle.jks.rdnsigncert.truststore.location =
rdn-sign-cert.p12
spring.ssl.bundle.jks.rdnsigncert.truststore.password =
#spring.ssl.bundle.pem.rdnsigncert.truststore.certificat
e = rdn-sign-cert.crt
#spring.ssl.bundle.pem.rdnsigncert.truststore.certificat
e = -----BEGIN CERTIFICATE-----....

# NOOTS IAM Hostname
noots.iam.hostname = iam.noots.gov.de
# NOOTS RDN Hostname
noots.rdn.hostname = rdn.noots.gov.de

# -----
# SAK-DC Anschlussprotokoll Einstellungen
# -----

# SAK-Server Adresse
server.address = localhost
# SAK-Server Port
server.port = 8443

# Optionale Basic-Auth Authentifizierung der
Schnittstellen des SAKs
# Schnittstellen des SAKs können laut der
Schnittstellenspezifikation optional via
# Basic-Auth gesichert werden. Sollte dies der Fall
sein, muss hier
# "basicauth.token.active" muss auf "true" gesetzt
```

```
werden, um den Token-Endpoint abzusichern.
basicauth.token.active = true
# "basicauth.other.active" muss auf "true" gesetzt
werden, um die Evidence- und Evidence-Offer-Endpunkte
mit Basic-Auth abzusichern.
basicauth.other.active = true
# Username + Passwort muss angegeben werden, sobald
mindestens einer der oberen Werte auf "true" gesetzt
sind.
# Sofern aktiviert, werden fuer die Basic
Authentifizierung, keine leeren Credentials
unterstuetzt.
basicauth.username =
basicauth.password =

# SAK-Server TLS
# Hardware-basierte Schlüsselspeicher (empfohlen)
spring.ssl.bundle.jks.server.keystore.type = PKCS11
spring.ssl.bundle.jks.server.keystore.password =
spring.ssl.bundle.jks.server.key.alias = sakdc-server-
cert
# Datei-basierte Schlüsselspeicher (nicht empfohlen,
bitte nur PKCS12 in diesem Fall verwenden)
#spring.ssl.bundle.jks.server.keystore.type = PKCS12
#spring.ssl.bundle.jks.server.keystore.location = sakdc-
server-cert.p12
#spring.ssl.bundle.jks.server.keystore.password =

# Separate TLS Server-Zertifikaten, falls der SAK unter
unterschiedlichen Hostnames exponiert ist
#server.ssl.server-name-bundles[0].server-name = sak-
dc1.example.com
#server.ssl.server-name-bundles[0].bundle = sakdc1
#spring.ssl.bundle.jks.sakdc1.keystore.type = PKCS11
#spring.ssl.bundle.jks.sakdc1.keystore.password =
#spring.ssl.bundle.jks.sakdc1.key.alias = sakdc1-server-
cert
#spring.ssl.bundle.jks.sakdc1.options.enabled-protocols
= ${sakdc.server.ssl.enabled-protocols}
#spring.ssl.bundle.jks.sakdc1.options.ciphers =
${sakdc.server.ssl.enabled-ciphers}

# SAK-Server TLS Client-Auth
# Einkommentieren, falls DCs ohne Client-Zertifikat den
SAK aufrufen
#server.ssl.client-auth = none
# Truststore mit vertrauenswürdigen Client-Cert-CAs
```

```
# Auskommentieren, falls DCs ohne Client-Zertifikat den
SAK aufrufen
spring.ssl.bundle.jks.server.truststore.type = PKCS12
spring.ssl.bundle.jks.server.truststore.location =
sakdc-server-truststore.p12
spring.ssl.bundle.jks.server.truststore.password =

# SAK-Server TLS deaktivieren (nicht empfohlen, bitte
sicherstellen, dass DCs nur über verschlüsselten Wege
den SAK aufrufen).
# ACHTUNG: 'server.port' wird standardmäßig auf 8080
gesetzt.
# In diesem Fall, alle 'spring.ssl.bundle.jks.server.*'
Properties auskommentieren.
#spring.profiles.active = plain-http

# -----
# SAK-DC Allgemeine Anwendungskonfiguration
# -----

# Arbeitsordner des Webservers
# Dieser muss für die Anwendung lesbar und beschreibbar
sein.
# Andere User und Anwendungen sollen nicht darauf
zugreifen können.
server.tomcat.basedir = /var/run/sak-dc

# Monitoring-Endpunkte der Anwendung
# Die folgenden Endpunkte werden über die URL
"http://{{management.server.address}}:{{management.serve
r.port}}/actuator/{{endpoint}}" auf dem
# konfigurierten Management-Port standardmäßig zur
Verfügung gestellt und können über den
Konfigurationswert
# "management.endpoints.web.exposure.include" selektiv
aktiviert und damit exponiert werden:
# - health: Der Health-Endpunkt stellt Informationen
über die Readiness und Liveness der Anwendung bereit
# - prometheus: Der Prometheus-Endpunkt stellt eine
Vielzahl an Metriken im Prometheus-Format zur Verfügung
# - info: Der Info-Endpunkt liefert interne
Informationen der Anwendung, z.B. Daten zum Build oder
zur aktuellen Laufzeitumgebung
# - sbom: Der SBOM-Endpunkt liefert eine Auflistung
aller Abhängigkeiten und Bibliotheken im SAK-DC
management.endpoints.web.exposure.include =
health,prometheus
```

```
# Verbindungseinstellungen für die Management-Endpoints
management.server.address = localhost
management.server.port = 9000

# Logging-Einstellungen
# Mögliche Log-Level: ERROR, WARN, INFO, DEBUG, TRACE
# In einer Produktivumgebung sollte immer min. WARN
gesetzt werden
# Basis-Level für alle Logmeldungen der Anwendung
# (spezifische Log-Level überschreiben die Basis)
logging.level.root = INFO

# Rate-Limiting
# Anzahl der Requests je Periode die erlaubt sind
resilience4j.ratelimiter.instances.globalRateLimiter.limitForPeriod = 50
# Periode in der die obige Anzahl an Requests erlaubt sind.
# Gültige Einheiten sind Sekunden, 's' und Nanosekunden, 'n'
resilience4j.ratelimiter.instances.globalRateLimiter.limitRefreshPeriod = 1s
```

Quellen

Tabelle 8. Quellen

Name der Quelle	Link	Version
Anschlusskonzept Data Consumer (AD-NOOTS-01)	https://gitlab.opencode.de/noots/public/ad-noots/noots-architektur/ad-noots-release/-/blob/main/AD-NOOTS-XX/AD-NOOTS-01_+Anschlusskonzept+Data+Consumer+_DC_.md	Release Februar 2025
Anschlusskonzept Data Provider (AD-NOOTS-02)	https://gitlab.opencode.de/noots/public/ad-noots/noots-architektur/ad-noots-release/-/blob/main/AD-NOOTS-XX/AD-NOOTS-02_+Anschlusskonzept+Data+Provider+_DP_.md	Release Februar 2025

Name der Quelle	Link	Version
Spring Boot Dokumentation ; Kapitel "Installing Spring Boot Applications"	https://docs.spring.io/spring-boot/3.5/how-to/deployment/installing.html	Version 3.5.5
Spring Boot Dokumentation ; Kapitel "Externalized Configuration"; Abschnitt "External Application Properties"	https://docs.spring.io/spring-boot/reference/features/external-config.html#features.external-config.files	Version 3.5.5
Spring Boot Dokumentation ; Kapitel "Prometheus (prometheus)"	https://docs.spring.io/spring-boot/api/rest/actuator/prometheus.html	Version 3.5.5
Kubernetes Dokumentation ; Kapitel "Pod Lifecycle"; Abschnitt "Types of probe"	https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle/#types-of-probe	