Sveriges Kommuner och Regioner

VÄGLEDNING

2026-06-15

Vårt dnr:
Klicka eller tryck här för att ange text.

Installationsvägledning KLASSA 5

Att identifiera alla de informationsmängder som finns inom en verksamhet kan vara svårt. Men genom att utgå från t.ex. de informationstillgångar eller system som en organisation använder (eller planerar att införa) kan man strukturera upp informationsmängderna och lättare hantera eventuella risker och hot.

KLASSA är ett självskattningsverktyg som fungerar som en länk mellan verksamhetens behov av skydd och tekniken som kan säkerställa detta skydd. Skydden kan vara såväl tekniska som organisatoriska och benämns ofta säkerhetsåtgärder (exempelvis i NIS2) eller skyddsåtgärder (exempelvis i GDPR).

Med hjälp av KLASSA:s kravkataloger och automatiserade flöden underlättas arbetet med att värdera de informationsmängder, system eller motsvarande tillgångar som en verksamhet har. Därmed blir det också enklare att skapa handlingsplaner med de identifierade säkerhetsåtgärder som behöver vidtas för att hålla verksamheten trygg, samt säkerställa relevant regelefterlevnad.

Genom att använda KLASSA kan verksamheten också sammanställa behovet av säkerhet avseende sina informationsmängder, inför upphandlingar av nya system eller verktyg.

Sveriges Kommuner och Regioner
info@skr.se, www.skr.se
Post: 118 82 Stockholm *Besök:* Hornsgatan 20
Tfn: 08-452 70 00 *Org nr:* 222000-0315

Innehåll

Installationsvägledning - Klassa 5 Onprem

Denna installationsvägledning kan endast tillämpas när Klassa körs som lokal installation, även kallad Klassa onprem.

För att förenkla Helm, en pakethanterare för Kubernetes, finns det också ett samlat underlag för konfigurationen i form av Klassa_onprem_Helm.

För användare och administratörer av Klassa finns ytterligare vägledning vilket inte minst är aktuellt för Klassa onprem.

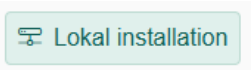
Alla vägledningar för Klassa finns här:
https://klassa.skr.se/sidor/stodmaterial/a-klassa-5-vagledning.

Licensiering

Ett separat avtal tecknas med Adda för att möjliggöra lokal installation av Klassa, onprem. När avtal är tecknat för onprem kommer en licensfil tillgängliggöras för organisationen.

Licensen kan sedan laddas ner från tjänsten Klassa (https://klassa.skr.se/online/) vilken kräver åtkomst till tjänsten. Om ni inte har åtkomst till tjänsten Klassa behöver det lösas först.

Väl inloggad, välj menyalternativet ”Lokal installation” som finns längst ner till vänster.



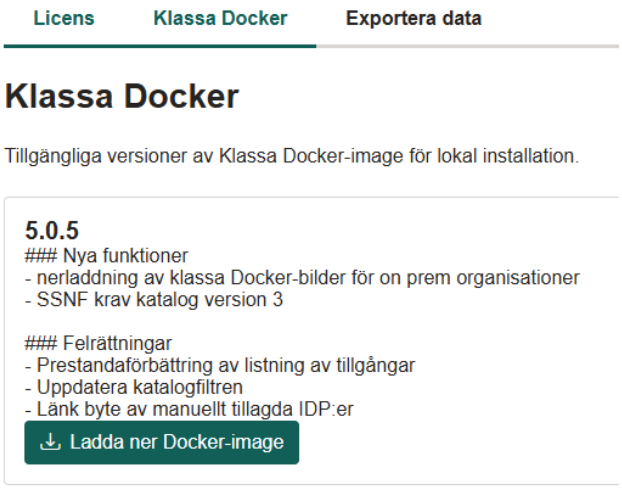
Detta menyalternativ finns bara om organisationen har onprem aktiverat.

Här laddas sedan licensfilen ner.

Licensfilen är en json-fil på ca 1kB.

Nedladdningar

Under samma menyalternativ som för licensieringen kan även Klassa Docker-image laddas ner.



Klassa Docker-image är en tar-fil på ca 400 MB.

För organisationer som önskar migrera data från Klassa version 4 finns ett menyalternativ benämnt ”Exportera data”. När ni beställer en migrering av data med stöd av Adda/SKR kommer de att göra en migrering av data från Klassa version 4 till Klassa version 5. Det datat går sedan att ladda ner från tjänsten KLASSA under menyalternativet ”Exportera data” som sedan importeras i Klassa onprem.

Lokala förutsättningar

Denna installationsvägledning exemplifierar en lokal installation av Klassa 5 med Ubuntu, PostgreSQL, Nginx och Kubernetes. Klassa 5 onprem kan fungera under andra förutsättningar men det är inte testat.

Komponent	Krav
Operativsystem	Ubuntu 24.04 LTS eller senare
RAM	Minst 4 GB
Disk	Minst 10 GB ledigt
Nätverk	Åtkomst till port 8080 (eller valfri port)

Installera Docker på Ubuntu

Steg 1: Ta bort eventuella gamla versioner

```
sudo apt remove docker.io docker-compose docker-compose-v2
docker-doc podman-docker containerd runc 2>/dev/null || true
```

Steg 2: Lägg till Dockers officiella APT-repository

```
sudo apt update
sudo apt install -y ca-certificates curl

sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o
/etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

sudo tee /etc/apt/sources.list.d/docker.sources <<EOF
Types: deb
URIs: https://download.docker.com/linux/ubuntu
Suites: $(. /etc/os-release && echo "${UBUNTU_CODENAME:-
$VERSION_CODENAME}")
Components: stable
Architectures: $(dpkg --print-architecture)
Signed-By: /etc/apt/keyrings/docker.asc
EOF

sudo apt update
```

Steg 3: Installera Docker Engine

```
sudo apt install -y docker-ce docker-ce-cli containerd.io
docker-buildx-plugin docker-compose-plugin
```

Steg 4: Verifiera installationen

```
sudo docker --version
```

Steg 5 (valfritt): Kör Docker utan sudo

```
sudo usermod -aG docker $USER
newgrp docker
```

Logga ut och in igen för att ändringen ska gälla permanent.

Installera jq

jq används av deployment-scriptet för att läsa `deployment-manifest.json`. Behövs inte om du startar Klassa manuellt med `docker run`.

```
sudo apt install -y jq
```

Databaskonfiguration

Klassa 5 kräver en PostgreSQL-databas. Välj ett av alternativen nedan.

Alternativ A: Lokal PostgreSQL i Docker

Enklaste alternativet för test och utveckling. Klassa-containern körs med `--network host` och når databasen via `localhost`.

```
docker run -d \
  --name klassa-db \
  --restart unless-stopped \
  -e POSTGRES_DB=klassa5 \
  -e POSTGRES_USER=klassa5 \
  -e POSTGRES_PASSWORD=<DITT_LÖSENORD> \
  -v klassa-pgdata:/var/lib/postgresql/data \
  -p 5432:5432 \
  postgres:15
```

Verifiera:

```
docker ps | grep klassa-db
```

Alternativ B: Extern PostgreSQL-server

1. Skapa databas och användare

```
CREATE DATABASE klassa5;
CREATE USER klassa5 WITH ENCRYPTED PASSWORD '<DITT_LÖSENORD>';
GRANT ALL PRIVILEGES ON DATABASE klassa5 TO klassa5;

-- PostgreSQL 15+ kräver även:
\c klassa5
GRANT ALL ON SCHEMA public TO klassa5;
```

2. Tillåt nätverksåtkomst

Filerna finns vanligtvis i `/etc/postgresql/<version>/main/` (Ubuntu/Debian) eller `/var/lib/pgsql/data/` (Red Hat/CentOS).

postgresql.conf — ändra `listen_addresses`:

```
listen_addresses = '*'
```

pg_hba.conf — lägg till en rad i slutet (ersätt `<KLASSA_SERVER_IP>`):

```
host    klassa5    klassa5    <KLASSA_SERVER_IP>/32    scram-sha-256
```

```
sudo systemctl restart postgresql
```

Starta Klassa

1. Ladda Docker-imagen

```
docker load -i klassa-<VERSION>.tar
```

Kontrollera att imagen laddades och notera taggen:

```
docker images | grep klassa
```

2. Förbered loggkatalog

```
sudo mkdir -p /opt/klassa/logs
sudo chmod 777 /opt/klassa/logs
```

3. Starta Klassa

Grundläggande start med användarnamn/lösenord-inloggning (ersätt `<TAG>` med taggen från steget ovan):

```
docker run -d \
  --name klassa-app \
  --restart unless-stopped \
  --network host \
  -e SPRING_PROFILES_ACTIVE=auth-password,production \
  -e SERVER_SERVLET_CONTEXT_PATH=/online \
  -e KLASSA_DB_URL=localhost:5432/klassa5 \
  -e KLASSA_DB_USER=klassa5 \
  -e KLASSA_DB_PWD=<DITT_LÖSENORD> \
  -v /opt/klassa/logs:/workspace/logs \
  klassa:<TAG>
```

*Alternativ — konfigurera via YAML-fil** i stället för miljövariabler: skapa en `application.yml` och montera den i containern:*
```bash
-v /opt/klassa/application.yml:/config/application.yml
`

4. Verifiera

```
docker ps | grep klassa-app
docker logs -f klassa-app
```

Öppna sedan en webbläsare och gå till:

```
http://<SERVER_IP>:8080/online/
```

Skapa första administratörsanvändaren

Klassa skapar ingen inloggningsbar standardanvändare vid uppstart. Första gången du kör med profilen `auth-password` måste du skapa en administratörsanvändare direkt i databasen.

*Förutsättning: ** Klassa måste ha startats minst en gång — Flyway-migrationer måste ha körts och skapat databasschemat.*

Steg 1: Generera BCrypt-lösenordshash

Installera `apache2-utils` om det saknas:

```
sudo apt-get install -y apache2-utils
```

Generera hash (ersätt `DittLösenord`):

```
htpasswd -nbB -C 10 dummy 'DittLösenord' | cut -d: -f2
```

Resultatet ser ut ungefär så här: `$2y$10$abc123...`

Steg 2: Skapa användaren i databasen

Anslut till databas-containern:

```
docker exec -it klassa-db psql -U klassa5 -d klassa5
```

Kör sedan följande SQL (ersätt e-post, namn och lösenordshash):

```
DO $$
DECLARE
  v_user_id UUID := gen_random_uuid();
  v_system  UUID := '9c8e7f6d-4a3b-2c1e-9f8e-7d6c5b4a3e2f';
  v_org     UUID := '3f7a8c2d-9e1b-4f6a-a8c3-7d2e9f1b4a6c';
BEGIN
  INSERT INTO klassa."user"
    (id, is_super_admin, owner_organization_id, password,
     created_by, created_date, last_modified_by,
last_modified_date)
  VALUES
    (v_user_id, true, v_org, '$2y$10$HASH_FRÅN_STEG_1',
     v_system, NOW(), v_system, NOW());

  INSERT INTO klassa.person
    (user_id, first_name, last_name, email,
owner_organization_id,
     created_by, created_date, last_modified_by,
last_modified_date)
  VALUES
    (v_user_id, 'Förnamn', 'Efternamn', 'admin@example.com',
     NULL, v_system, NOW(), v_system, NOW());

  INSERT INTO klassa.user_organization_roles (user_id,
organization_roles)
  VALUES (v_user_id, 'ADMIN');
END $$;
```

> *`is_super_admin = true`** ger användaren åtkomst till alla organisationer och Klassa-administratörsvyn. Rollen kan tas bort i GUI:n efter att ytterligare administratörer skapats.*

Steg 3: Logga in

Navigera till `/online/login/password` och logga in med den e-postadress och det lösenord du angav ovan.

Licensiering

Klassa kräver en licensfil (`.json`) som laddas ner från KLASSA-tjänsten (se ovan). Filen kan ligga var som helst på servern så länge Klassa-processen har läsrättighet på den. Rekommenderad sökväg: `/opt/klassa/licens/klassa-license-<org>.json`.

Du monterar in licensfilen (eller dess katalog) som en read-only volym — på exakt samma sätt som SAML-certifikaten — och pekar sedan ut den med antingen en miljövariabel eller via `application.yml`.

Variabel	Standard	Beskrivning
`KLASSA_LICENSE_CERT_FILE_PATH`	**(obligatorisk)**	Sökväg inuti containern till licensfilen

Alternativ 1 — Miljövariabel (rekommenderat för Docker/systemd)

Lägg till `-v`-flaggan och miljövariabeln i din `docker run` (samma container som startas i [Starta Klassa](#starta-klassa)):

```
docker run -d \
  --name klassa-app \
  --restart unless-stopped \
  --network host \
  -e SPRING_PROFILES_ACTIVE=auth-password,production \
  -e SERVER_SERVLET_CONTEXT_PATH=/online \
  -e KLASSA_DB_URL=localhost:5432/klassa5 \
  -e KLASSA_DB_USER=klassa5 \
  -e KLASSA_DB_PWD=<DITT_LÖSENORD> \
  -e KLASSA_LICENSE_CERT_FILE_PATH=/opt/klassa/licens/klassa-
license<org>.json \
  -v /opt/klassa/logs:/workspace/logs \
  -v /opt/klassa/licens:/opt/klassa/licens:ro \
  klassa:<TAG>
```

Alternativ 2 — `application.yml`

```
klassa:
  license:
    cert-file-path: /opt/klassa/licens/klassa-license-<org>.json
```

OBS: Sökvägen efter kolon i `-v`-flaggan (container-sökvägen) måste matcha den du anger i `KLASSA_LICENSE_CERT_FILE_PATH` / `cert-file-path`. Sökvägen före kolon (sökvägen på värden) kan vara vad som helst.

Vid uppstart ska du se loggraden `License verified: organization=…`. Saknas licensen eller är ogiltig stängs Klassa ner med ett felmeddelande i loggen.

Spring-profiler

Aktiveras via miljövariabeln `SPRING_PROFILES_ACTIVE`. Kombinera med kommatecken.

Profil	Funktion
`auth-password`	Användarnamn/lösenord-inloggning
`auth-managed`	SAML-inloggning med IdP-konfiguration via GUI. SP-certifikat anges via miljövariabler.
`auth-saml`	Intern SAML-inloggning (SKR) med statisk IdP-konfiguration i YAML/miljövariabler.
`auth-bank-id`	BankID-inloggning via SAML. Kräver `auth-saml`.
`auth-federation`	Federerad SAML (Internetstiftelsen). Kräver `auth-saml`.
`notification-email`	Utgående e-postnotifieringar
`production`	Döljer "Testmiljö"-bannern i gränssnittet

Exempel:

```
SPRING_PROFILES_ACTIVE=auth-password,auth-managed,production
```

Miljövariabler — referens

Databas

Variabel	Standard	Beskrivning
`KLASSA_DB_URL`	`localhost:5432/postgres`	PostgreSQL `host:port/dbnamn`
`KLASSA_DB_USER`	`postgres`	Databasanvändare
`KLASSA_DB_PWD`	`postgres`	Databaslösenord

Server

Variabel	Standard	Beskrivning
`KLASSA_APP_PORT`	`8080`	HTTP-port
`SERVER_SERVLET_CONTEXT_PATH`	`/online`	Context path — **måste vara `/online`** annars genereras felaktiga redirect-URL:er
`TOMCAT_ACCESSLOG_DIR`	`logs`	Katalog för Tomcat-åtkomstloggar

Allmänt

Variabel	Standard	Beskrivning
`KLASSA_AUTH_DEFAULT_METHOD`	`discovery`	Vart icke-inloggade användare skickas. Värden: `discovery` (väljarvy), `password`, `internal-saml`, `bank-id`

Fullständigt exempel — application.yml

I stället för miljövariabler kan du montera en `application.yml` i containern på `/config/application.yml`:

```
spring:
  profiles:
    active: auth-password,auth-saml,auth-bank-id,auth-
federation,notification-email,production

  datasource:
    url: jdbc:postgresql://db-host:5432/klassa5
    username: klassa5
    password: <LÖSENORD>

  mail:
    host: smtp.example.com
    port: 587
    username: user
    password: secret

  security:
    saml2:
      relyingparty:
        registration:
          saml:
            entity-id:
https://klassa.example.com/saml2/service-provider-metadata/saml
            signing:
              credentials:
                - private-key-location:
file:/opt/klassa/certificates/saml.key
                  certificate-location:
file:/opt/klassa/certificates/saml.crt
            assertingparty:
              metadata-uri: https://idp.example.com/metadata
          bank-id:
            entity-id:
https://klassa.example.com/saml2/service-provider-
metadata/bank-id
            signing:
              credentials:
                - private-key-location:
file:/opt/klassa/certificates/bank-id.key
                  certificate-location:
file:/opt/klassa/certificates/bank-id.crt
            assertingparty:
              metadata-uri: https://bankid-
idp.example.com/metadata
          internetstiftelsen:
```

```
            entity-id:
https://klassa.example.com/saml2/service-provider-
metadata/internetstiftelsen
            signing:
              credentials:
                - private-key-location:
file:/opt/klassa/certificates/internetstiftelsen.key
                  certificate-location:
file:/opt/klassa/certificates/internetstiftelsen.crt

server:
  port: 8080
  tomcat:
    accesslog:
      directory: /workspace/logs

klassa:
  auth:
    default-method: discovery   # discovery | password |
internal-saml | bank-id

  saml:
    internal:
      visible: true
      display-name: "Sveriges Kommuner och Regioner"
      path: saml
    bankid:
      visible: true
      display-name: "BankID"
      path: bank-id
    federated: {}
    sp:
      metadata:
        organization-name: <ER_ORGANISATION>
        organization-url: <ER_WEBBPLATS>
        administrative-email: <ADMIN_EPOST>
        technical-email: <TEKNISK_EPOST>
        support-email: <SUPPORT_EPOST>
        service-name: Klassa

  internetstiftelsens-federationer:
    metadata-url:
https://md.openfed.se/prod/md/metadata_set1_idp_01.xml
    signing-certificate-location:
file:/opt/klassa/certificates/internetstiftelsens-federationer-
metadata.crt
    template-registration-id: internetstiftelsen

  notification:
    from-address: noreply@klassa.se
    base-url: https://<DIN_DOMÄN>
```

SAML attribut

Krävda IdP-attribut

IdP:n måste skicka följande attribut i SAML-assertionen. Klassa använder dem för att identifiera användaren, fylla i profilen och tilldela roller och organisation.

Attributnamn	Friendly name	Källa (LDAP/AD)	Obligatorisk	Beskrivning
`uid`	uid	`uid`	Ja	Unikt användar-ID — används som Klassa-användarnamn
`mail`	mail	`mail`	Ja	E-postadress
`gn`	gn	`givenName`	Ja	Förnamn
`sn`	sn	`sn`	Ja	Efternamn
`OrgID`	OrgID	`department`	Ja	Organisations-ID — kopplar användaren till en organisation i Klassa
`Roles`	Roles	`memberOf`	Nej	Gruppmedlemskap — används för rolltilldelning i Klassa

*OBS:** Attributnamnen är skiftlägeskänsliga. Se till att IdP:n skickar dem exakt som ovan (t.ex. `OrgID`, inte `orgid`).*

Hanterad SAML — Profil: `auth-managed`

Med `auth-managed` konfigureras IdP:n i Klassas GUI efter uppstart. Du behöver bara tillhandahålla SP-signeringscertifikatet via miljövariabler.

Generera SP-certifikat

```
sudo mkdir -p /opt/klassa/managed
sudo openssl req -x509 -newkey rsa:2048 \
  -keyout /opt/klassa/managed/saml.key \
  -out /opt/klassa/managed/saml.crt \
  -days 3650 -nodes -subj "/CN=klassa"
sudo chmod 644 /opt/klassa/managed/saml.key
```

> *OBS:** Certifikatfiler som monteras in i containern måste ha rättighet `644` (läsbar av alla) — containerns process-user är inte root och kan inte läsa `600`-filer.*

Starta Klassa med `auth-managed`

```
docker run -d \
  --name klassa-app \
  --restart unless-stopped \
  --network host \
  -e SPRING_PROFILES_ACTIVE=auth-password,auth-
managed,production \
  -e SERVER_SERVLET_CONTEXT_PATH=/online \
  -e KLASSA_DB_URL=localhost:5432/klassa5 \
  -e KLASSA_DB_USER=klassa5 \
  -e KLASSA_DB_PWD=<DITT_LÖSENORD> \
  -e KLASSA_BASE_URL=https://<DIN_DOMÄN> \
  -e
KLASSA_SAML_MANAGED_KEY_LOCATION=file:/opt/klassa/managed/saml.
key \
  -e
KLASSA_SAML_MANAGED_CERT_LOCATION=file:/opt/klassa/managed/saml
.crt \
  -v /opt/klassa/logs:/workspace/logs \
  -v
/opt/klassa/managed/saml.key:/opt/klassa/managed/saml.key:ro \
  -v
/opt/klassa/managed/saml.crt:/opt/klassa/managed/saml.crt:ro \
  klassa:<TAG>
```

Miljövariabler för `auth-managed`

Variabel	Beskrivning
`KLASSA_BASE_URL`	Applikationens publika bas-URL — används i SP-metadata, ACS-URL och e-postnotifieringslänkar
`KLASSA_SAML_MANAGED_KEY_LOCATION`	Spring-resurssökväg till SP-signeringsnyckeln, t.ex. `file:/opt/klassa/managed/saml.key`
`KLASSA_SAML_MANAGED_CERT_LOCATION`	Spring-resurssökväg till SP-signeringscertifikatet, t.ex. `file:/opt/klassa/managed/saml.crt`

Certifikatvolym

SAML kräver SP-certifikat för signering. Standardsökvägarna utgår från att certifikaten är monterade på `/opt/klassa/certificates/` i containern:

```
/opt/klassa/certificates/
  saml.key / saml.crt                                ← Intern
SAML SP-nyckelpar
  bank-id.key / bank-id.crt                          ← BankID
SP-nyckelpar
  internetstiftelsen.key / internetstiftelsen.crt    ← ISF
federerad SP-nyckelpar
  internetstiftelsens-federationer-metadata.crt      ← ISF
metadata-signeringscertifikat
```

Montera katalogen som en Docker-volym:

```
-v /opt/klassa/certificates:/opt/klassa/certificates:ro
```

Intern SAML (SKR) — Profil: `auth-saml`

Variabel	Standard	Beskrivning
`KLASSA_SAML_INTERNAL_ENTITY_ID`	`http://localhost:8080/saml2/service-provider-metadata/saml`	SP Entity ID
`KLASSA_SAML_INTERNAL_METADATA_URI`	*(obligatorisk)*	IdP metadata-URL
`KLASSA_SAML_KEY_LOCATION`	`file:/opt/klassa/certificates/saml.key`	SP signeringsnyckel

KLASSA_SAML_CERT_LOCATION	file:/opt/klassa/certificates/saml.crt	SP signeringscertifikat
KLASSA_SAML_INTERNAL_VISIBLE	true	Visa i väljarvy
KLASSA_SAML_INTERNAL_DISPLAY_NAME	Sveriges Kommuner och Regioner	Etikett i väljarvy
KLASSA_SAML_INTERNAL_PATH	saml	Inloggnings-URL (/login/saml)

BankID — Profil: `auth-bank-id` + `auth-saml`

Variabel	Standard	Beskrivning
KLASSA_SAML_BANKID_ENTITY_ID	http://localhost:8080/saml2/service-provider-metadata/bank-id	SP Entity ID
KLASSA_SAML_BANKID_METADATA_URI	*(obligatorisk)*	IdP metadata-URL
KLASSA_SAML_BANKID_KEY_LOCATION	file:/opt/klassa/certificates/bank-id.key	SP signeringsnyckel
KLASSA_SAML_BANKID_CERT_LOCATION	file:/opt/klassa/certificates/bank-id.crt	SP signeringscertifikat
KLASSA_SAML_BANKID_VISIBLE	true	Visa i väljarvy
KLASSA_SAML_BANKID_DISPLAY_NAME	BankID	Etikett i väljarvy
KLASSA_SAML_BANKID_PATH	bank-id	Inloggnings-URL (/login/bank-id)

Federerad SAML (ISF) — Profil: `auth-federation` + `auth-saml`

Variabel	Standard	Beskrivning
KLASSA_SAML_FEDERATED_ENTITY_ID	http://localhost:8080/saml2/service-provider-metadata/internetstiftelsen	SP Entity ID
KLASSA_SAML_FEDERATED_KEY_LOCATION	file:/opt/klassa/certificates/internetstiftelsen.key	SP signeringsnyckel
KLASSA_SAML_FEDERATED_CERT_LOCATION	file:/opt/klassa/certificates/internetstiftelsen.crt	SP signeringscertifikat
KLASSA_SAML_FEDERATED_METADATA_URLS	*(tom)*	Kommaseparerade IdP metadata-URL:er
KLASSA_SAML_FEDERATED_CERT_URLS	*(tom)*	Kommaseparerade metadata-signeringscertifikat-URL:er
KLASSA_SAML_FEDERATED_SIGNING_CERT_LOCATION	file:/opt/klassa/certificates/internetstiftelsens-federationer-metadata.crt	Metadata-signeringscertifikat (fil)
KLASSA_SAML_FEDERATED_TEMPLATE_REGISTRATION_ID	internetstiftelsen	Mall-registrerings-ID

SP-metadata — Profil: `auth-saml`

Används vid federationsregistrering — organisationsinformation som visas i IdP:ns väljarvy.

Variabel	Standard	Beskrivning
KLASSA_SAML_SP_ORGANIZATION_NAME	Adda AB	Organisationsnamn i metadata
KLASSA_SAML_SP_ORGANIZATION_URL	https://adda.se/	Organisations-URL
KLASSA_SAML_SP_ADMINISTRATIVE_EMAIL	adda@regent.se	Administrativ kontakt
KLASSA_SAML_SP_TECHNICAL_EMAIL	adda@regent.se	Teknisk kontakt
KLASSA_SAML_SP_SUPPORT_EMAIL	adda@regent.se	Supportkontakt
KLASSA_SAML_SP_SERVICE_NAME	Klassa	Tjänstenamn
KLASSA_SAML_SP_SERVICE_DESCRIPTION_SV	Ett verktyg som hjälper organisationer att systematiskt arbeta med informationssäkerhet	Tjänstebeskrivning (SV)
KLASSA_SAML_SP_SERVICE_DESCRIPTION_EN	A tool that helps organizations systematically work with information security	Tjänstebeskrivning (EN)
KLASSA_SAML_SP_MDUI_DISPLAY_NAME	Klassa	MDUI visningsnamn
KLASSA_SAML_SP_MDUI_LOGO_URL	/online/img/logo_sv.svg	MDUI logotyp-URL
KLASSA_SAML_SP_MDUI_LOGO_WIDTH	320	Logotypbredd (px)
KLASSA_SAML_SP_MDUI_LOGO_HEIGHT	132	Logotyphöjd (px)

E-postnotifieringar

Profil: `notification-email`

Variabel	Standard	Beskrivning
`KLASSA_MAIL_HOST`	`localhost`	SMTP-server
`KLASSA_MAIL_PORT`	`1025`	SMTP-port
`KLASSA_MAIL_USERNAME`	*(tom)*	SMTP-användarnamn
`KLASSA_MAIL_PASSWORD`	*(tom)*	SMTP-lösenord
`KLASSA_MAIL_FROM`	`noreply@klassa.se`	Avsändaradress
`KLASSA_BASE_URL`	`http://localhost:8080`	Applikationens publika bas-URL — används i notifieringslänkar och av `auth-managed` för SP-metadata/ACS-URL

Reverse proxy — exempel med nginx

Klassa hanterar inte TLS och ska alltid köras bakom en reverse proxy som terminerar HTTPS. Det finns många alternativ — Caddy, Traefik, HAProxy, Apache m.fl. Nedan visas nginx som ett konkret exempel.

Installera nginx

```
sudo apt install -y nginx
```

Konfigurationsfil

Skapa `/etc/nginx/sites-available/klassa`:

```
# HTTP → HTTPS-redirect
server {
    listen 80;
    server_name <DIN_DOMÄN>;
    return 301 https://$host$request_uri;
}

# HTTPS
server {
    listen 443 ssl;
    server_name <DIN_DOMÄN>;

    ssl_certificate     /etc/nginx/certs/fullchain.pem;
    ssl_certificate_key /etc/nginx/certs/privkey.pem;
    ssl_protocols       TLSv1.2 TLSv1.3;
    ssl_ciphers         HIGH:!aNULL:!MD5;

    location /online/ {
        proxy_pass         http://localhost:8080/online/;
        proxy_http_version 1.1;
        proxy_set_header   Host              $host;
        proxy_set_header   X-Real-IP         $remote_addr;
        proxy_set_header   X-Forwarded-For
$proxy_add_x_forwarded_for;
        proxy_set_header   X-Forwarded-Proto $scheme;

        # Krävs för att SAML-redirectar ska generera rätt
URL:er
        proxy_redirect     http://localhost:8080/
https://$host/;
    }

    # Omdirigera rooten till /online/
    location = / {
        return 302 /online/;
    }
}
```

*OBS:** `X-Forwarded-Proto: https` är viktigt för SAML — utan det genererar Klassa HTTP-baserade ACS-URL:er som IdP:n avvisar.*

Aktivera och starta

```
sudo mkdir -p /etc/nginx/certs
sudo cp fullchain.pem /etc/nginx/certs/
sudo cp privkey.pem   /etc/nginx/certs/
sudo chmod 600 /etc/nginx/certs/privkey.pem

sudo ln -sf /etc/nginx/sites-available/klassa /etc/nginx/sites-
enabled/klassa
sudo rm -f /etc/nginx/sites-enabled/default
sudo nginx -t && sudo systemctl reload nginx
```

*OBS:** `default`-siten inaktiveras så att nginx inte svarar med en annan konfiguration på port 80/443.*

Felsökning

Visa loggar

```
docker logs -f klassa-app
```

Kontrollera containerstatus

```
docker ps -a | grep klassa
```

Starta om Klassa

```
docker restart klassa-app
```

Stoppa Klassa

```
docker stop klassa-app
```

Driftsätt Klassa i Kubernetes

Denna sektion beskriver hur du driftsätter Klassa 5 i ett befintligt Kubernetes-kluster med hjälp av manifesten i mappen `kubernetes/`.

Förutsättningar

Komponent	Krav
Kubernetes	v1.25 eller senare
Storage	StorageClass med dynamisk provisionering (t.ex. `local-path`)
kubectl	Konfigurerad med åtkomst till klustret (`~/.kube/config`)
TLS-terminering	Ingress-controller (Traefik, HAProxy, nginx m.fl.) eller extern reverse proxy — Klassa hanterar inte TLS

> *Viktigt:** Kontrollera att följande systempoddar körs innan du börjar:*
> ```bash
> *kubectl get pods -n kube-system | grep -E 'coredns|local-path'*
> ```
> *Om CoreDNS saknas:* `sudo kubeadm init phase addon coredns`
> *Om local-path-provisioner saknas: se [Felsökning Kubernetes](#felsökning-kubernetes) nedan.*

Steg 1: Ladda imagen på varje nod

Kubernetes använder containerd direkt och hämtar inte imagen från Docker — den måste importeras på **varje nod** (master + workers).

Kopiera tar-filen till respektive nod:

```
scp klassa-<VERSION>.tar ubuntu@<MASTER_IP>:/tmp/
scp klassa-<VERSION>.tar ubuntu@<WORKER1_IP>:/tmp/
scp klassa-<VERSION>.tar ubuntu@<WORKER2_IP>:/tmp/
```

Importera på **varje nod** (kör via SSH):

```
sudo ctr -n k8s.io images import /tmp/klassa-<VERSION>.tar
```

Verifiera att imagen finns och notera den exakta taggen:

```
sudo ctr -n k8s.io images ls | grep klassa
```

Uppdatera sedan `image`-fältet i `kubernetes/deployment.yaml` med den exakta taggen (t.ex. `docker.io/library/klassa:7678`).

Steg 2: Konfigurera manifesten

`kubernetes/secret-db.yaml`

```
stringData:
  db-password: "<DITT_DATABASLÖSENORD>"
```

`kubernetes/postgres.yaml`

```
stringData:
  POSTGRES_PASSWORD: "<DITT_DATABASLÖSENORD>"
```

Lösenordet måste vara identiskt i båda filerna.

`kubernetes/deployment.yaml`

Sätt rätt image och justera vid extern PostgreSQL-server:

```
image: <REGISTRY>/klassa:<TAG>
```

```
env:
  - name: KLASSA_DB_URL
    value: "<POSTGRES_HOST>:5432/klassa5"
  - name: KLASSA_DB_USER
    value: "klassa5"
  - name: KLASSA_NOTIFICATION_BASE_URL
    value: "https://<DIN_DOMÄN>/online"
```

`kubernetes/ingress.yaml`

Klassa exponerar HTTP på port 80 — TLS-terminering hanteras av ingress-controllern.

```
tls:
  - hosts:
      - <DIN_DOMÄN>
    secretName: klassa-tls
rules:
  - host: <DIN_DOMÄN>
```

Om klustret inte har en standardklass för ingress, lägg till `ingressClassName` under `spec:` i filen:

```
spec:
  ingressClassName: <DIN_INGRESS_KLASS>   # t.ex. traefik,
haproxy
```

Steg 3: Skapa TLS-secret

Behövs när TLS hanteras av en ingress-controller i klustret. Hoppa över detta steg om du använder en extern reverse proxy.

```
kubectl create secret tls klassa-tls \
  --cert=<sökväg-till-fullchain.pem> \
  --key=<sökväg-till-private.key> \
  -n klassa
```

Steg 4: Driftsätt

```
kubectl apply -f kubernetes/namespace.yaml
kubectl apply -f kubernetes/secret-db.yaml
kubectl apply -f kubernetes/postgres.yaml
kubectl apply -f kubernetes/configmap.yaml
kubectl apply -f kubernetes/pvc-logs.yaml
kubectl apply -f kubernetes/deployment.yaml
kubectl apply -f kubernetes/service.yaml
kubectl apply -f kubernetes/ingress.yaml
```

Steg 5: Verifiera

```
kubectl get pods -n klassa -o wide
```

Förväntat resultat (efter ~2 minuter):

```
NAME                       READY   STATUS    RESTARTS   AGE
klassa-xxxxxxxxx-xxxxx     1/1     Running   0          2m
postgres-0                 1/1     Running   0          2m
```

```
kubectl logs -f deployment/klassa -n klassa
kubectl get ingress -n klassa
```

Klassa är tillgängligt på `https://<DIN_DOMÄN>/online/`.

Felsökning Kubernetes

Poddar fastnar i `Pending`

```
kubectl get pods -n local-path-storage
```

Om provisionern saknas:

```
kubectl apply -f
https://raw.githubusercontent.com/rancher/local-path-
provisioner/v0.0.28/deploy/local-path-storage.yaml
```

DNS-fel (`UnknownHostException: postgres`)

```
kubectl get pods -n kube-system | grep coredns
```

Om CoreDNS saknas:

```
sudo kubeadm init phase addon coredns
kubectl rollout restart deployment/klassa -n klassa
```

`404 Not Found`

Kontrollera att du använder rätt URL — Klassa svarar på `/online/`:

```
https://<DIN_DOMÄN>/online/
```

Felsöka en kraschad pod

```
kubectl logs -n klassa deployment/klassa --previous
kubectl describe pod -n klassa <POD_NAMN>
```

Starta om Klassa

```
kubectl rollout restart deployment/klassa -n klassa
```

Driftsätt Klassa med Helm

Helm är ett pakethanteringssystem för Kubernetes som samlar alla manifest i ett *chart*. I stället för att köra flera `kubectl apply`-kommandon installeras och uppgraderas allt med ett enda kommando. Helm håller även versionshistorik och kan återgå till en tidigare version vid behov.

För att förenkla Helm finns det också ett samlat underlag för konfigurationen i form av Klassa_onprem_Helm.

Helm och kubectl körs från din **lokala arbetsstation** — du behöver inte logga in på någon nod.

Förutsättningar

Komponent	Krav
Helm	v3.x (`helm version`)
kubectl	Konfigurerad med åtkomst till klustret (`kubectl get nodes`)
Kubernetes	v1.25 eller senare

Installera Helm om det saknas (Linux/Mac):

```
curl
https://raw.githubusercontent.com/helm/helm/main/scripts/get-
helm-3 | bash
```

På Windows — ladda ned binären från helm.sh/docs/intro/install och lägg den i PATH.

Chartets struktur

Chartet finns i mappen `helm/klassa/`. All konfiguration styrs via `helm/klassa/values.yaml` — du behöver inte redigera YAML-mallarna direkt.

Steg 1: Uppdatera image-tagg

Öppna `helm/klassa/values.yaml` och sätt `image.tag` till den exakta taggen på tar-filen:

```
image:
  tag: "7804"
```

Steg 2: Ladda imagen på varje nod

> *Gäller inte Docker Desktop** — där används Docker-daemonen direkt och bilden laddas med `docker load`.*

Kopiera tar-filen till varje nod och importera den:

```
sudo ctr -n k8s.io images import /tmp/klassa-<VERSION>.tar
```

Upprepa på samtliga noder (master + workers).

Steg 3: Skapa namespace

Namespacet måste finnas innan secrets skapas:

```
kubectl create namespace klassa
```

Steg 4: Skapa TLS- och SAML-secrets

> *Hoppa över detta steg** om du använder en extern reverse proxy (nginx m.m.) som hanterar TLS — sätt då `ingress.enabled=false` i steg 5 i stället.*

Helm hanterar inte certifikat — dessa skapas en gång manuellt:

```
# TLS-certifikat för Ingress
kubectl create secret tls klassa-tls \
  --cert=fullchain.pem \
  --key=privkey.pem \
  -n klassa --dry-run=client -o yaml | kubectl apply -f -
```

```
# SAML-certifikat (endast om auth-saml används)
kubectl create secret generic klassa-saml-certs \
  --from-file=saml.key \
  --from-file=saml.crt \
  -n klassa --dry-run=client -o yaml | kubectl apply -f -
```

Steg 5: Installera

Med extern reverse proxy (ingen Ingress):

Används när nginx eller liknande hanterar TLS utanför klustret. Klassa exponeras på port 8080 direkt på varje nods IP via `hostPort`.

```
helm install klassa ./helm/klassa \
  --set db.password=<DITT_LÖSENORD> \
  --set domain=<DIN_DOMÄN> \
  --set ingress.enabled=false \
  --set service.hostPort=8080
```

Din nginx proxar sedan till `http://<NOD-IP>:8080/online/`.

Med Ingress-controller i klustret:

Klassa exponerar HTTP på port 80 — TLS-terminering hanteras av ingress-controllern. Ange `ingress.className` om klustret inte har en standardklass:

```
helm install klassa ./helm/klassa \
  --set db.password=<DITT_LÖSENORD> \
  --set domain=<DIN_DOMÄN> \
  --set ingress.className=<DIN_INGRESS_KLASS>
```

`ingress.className` kan utelämnas om klustret har en standardklass konfigurerad.

Med SAML aktiverat (lägg till oavsett ingress-variant):

```
  --set spring.profiles=auth-password,auth-saml,production \
  --set
saml.internal.metadataUri=https://idp.example.com/metadata
```

> *Tips:** Skapa en egen `my-values.yaml` med dina inställningar i stället för långa `--set`-kedjor och kör `helm install klassa ./helm/klassa -f my-values.yaml`. Det förenklar även uppgraderingar.*

Steg 6: Verifiera

```
helm status klassa
kubectl get pods -n klassa
kubectl logs -f deployment/klassa -n klassa
```

Klassa är tillgängligt på `https://<DIN_DOMÄN>/online/`.

Uppgradera till ny version

Ladda först in den nya imagen på varje nod (steg 2), sedan:

```
helm upgrade klassa ./helm/klassa --reuse-values --set
image.tag=<NY_TAG>
```

`--reuse-values` bevarar alla tidigare inställningar (lösenord, domän m.m.) — du behöver bara ange det som ändrats.

Återgå till föregående version

```
helm history klassa          # visa versionshistorik
helm rollback klassa 1       # återgå till revision 1
```

Avinstallera

```
helm uninstall klassa
```

> *OBS: **PersistentVolumeClaims (databas och loggar) tas** inte** bort automatiskt vid avinstallation. Ta bort dem manuellt om du vill rensa upp:*
> *```bash*
> *kubectl delete pvc -n klassa --all*
> *```*

Extern PostgreSQL

Sätt `postgres.enabled=false` och ange `db.host` för att använda en befintlig databasserver:

```
helm install klassa ./helm/klassa \
  --set db.password=<LÖSENORD> \
  --set db.host=<POSTGRES_HOST> \
  --set postgres.enabled=false \
  --set domain=<DIN_DOMÄN>
```

Säkerhetshärdning

Det här avsnittet ger en övergripande vägledning för att säkra upp den miljö där Klassa körs. Exakta kommandon och konfigurationsformat varierar beroende på Linux-distribution, containerplattform och webbserver — betrakta detta som en checklista snarare än en steg-för-steg-guide. Det är heller inte en uttömmande guide för härdning av er Klassa onprem.

Operativsystem

- **Håll systemet uppdaterat.** Aktivera automatiska säkerhetsuppdateringar eller etablera en rutin för regelbunden patchning. Det gäller både värdoperativet och eventuella container-images.
- **Minimera exponerade tjänster.** Stäng portar och tjänster som inte behövs. Använd en brandvägg (t.ex. `ufw`, `firewalld` eller nftables) och tillåt endast trafik som är nödvändig — typiskt SSH från betrodda adresser, HTTP/HTTPS mot omvärlden, och databasport endast internt.
- **Begränsa SSH-åtkomst.** Inaktivera lösenordsinloggning och root-inloggning via SSH. Använd SSH-nycklar och begränsa vilka användare och adresser som får logga in.
- **Använd separata, begränsade användarkonton.** Kör aldrig applikationer som root om det kan undvikas. Skapa dedikerade tjänstekonton med minimal behörighet.
- **Aktivera och granska loggning.** Se till att systemloggar (auth, kernel, tjänster) samlas in och bevaras. Överväg central logghantering.

Obligatorisk åtkomstkontroll — SELinux och AppArmor

Moderna Linux-distributioner levereras med stöd för obligatorisk åtkomstkontroll (MAC) som begränsar vad processer får göra även om de kompromitteras.

- **SELinux** används primärt på RHEL-baserade system (Rocky Linux, AlmaLinux, Fedora m.fl.). Kör i `enforcing`-läge i produktion — `permissive` loggar men skyddar inte.
- **AppArmor** används primärt på Debian/Ubuntu. Aktivera profiler för tjänster som exponeras mot nätverk.

Inaktivera inte MAC-skyddet för att lösa konfigurationsproblem — rätt åtgärd är att justera policyn. Containermiljöer som Docker och Kubernetes har inbyggt stöd för att köra med SELinux- respektive AppArmor-profiler.

Docker och containersäkerhet

- **Kör inte containrar som root.** Sätt `user` i docker-compose eller i Dockerfile. Klassa-imagen är konfigurerad för att köra som icke-privilegierad användare.
- **Begränsa containerns resurser.** Sätt minnesgränser (`--memory`) och CPU-kvoter för att begränsa effekten av en eventuell kompromiss.
- **Använd `--read-only` och begränsade capabilities** där möjligt. Montera bara de kataloger containern faktiskt behöver.
- **Exponera aldrig Docker-sockeln** (`/var/run/docker.sock`) i en container utan stark motivering — det ger i praktiken root-åtkomst till värden.
- **Skanna images regelbundet** efter kända sårbarheter med verktyg som Trivy, Grype eller Docker Scout.
- **Håll base images uppdaterade.** Bygg om och omdeploya när en ny säkerhetskorrigerad bas-image finns tillgänglig.

Reverse proxy och HTTPS

All trafik till Klassa bör gå via en reverse proxy som terminerar TLS. Klassa själv ska aldrig exponeras direkt mot internet utan TLS.

- **Använd ett giltigt certifikat** utfärdat av en betrodd CA. Let's Encrypt via Certbot eller ACME-klienter är ett vanligt val för automatisk förnyelse. Interna deployments kan använda organisationens interna PKI.
- **Konfigurera automatisk certifikatförnyelse.** Certifikat från Let's Encrypt löper ut efter 90 dagar — säkerställ att förnyelsen testas och fungerar i god tid.
- **Tillåt enbart moderna protokoll och krypteringssviter.** Inaktivera TLS 1.0 och 1.1. Använd TLS 1.2 som minimum, helst TLS 1.3. Referera till [Mozilla SSL Configuration Generator](https://ssl-config.mozilla.org/) för rekommenderade inställningar.
- **Sätt säkerhetsrelaterade HTTP-huvuden** i proxyn: `Strict-Transport-Security`, `X-Content-Type-Options`, `X-Frame-Options` och `Content-Security-Policy`.
- **Begränsa vilka IP-adresser** som kan nå proxyn om Klassa bara ska vara tillgänglig från ett specifikt nät.
- **Säkra proxytjänsten i sig** — håll nginx, Caddy, HAProxy eller annat verktyg uppdaterat och exponera inte administrationsgränssnitt mot omvärlden.

Databas

- **Exponera aldrig databasportén mot internet.** PostgreSQL ska endast vara åtkomlig från applikationsservern eller Kubernetes-nätverket.
- **Använd starka, unika lösenord** för databasanvändaren. Lagra dem aldrig i klartext i konfigurationsfiler som checkas in i versionshantering.
- **Begränsa databasanvändarens rättigheter** till enbart den databas Klassa använder — undvik superuser-rättigheter.
- **Aktivera krypterad anslutning** (SSL/TLS) mellan applikation och databas om de kommunicerar över ett nätverk som inte är helt betrodd.
- **Ta regelbundna säkerhetskopior** och testa återläsning. Förvara säkerhetskopior på separat plats från produktionsmiljön.

Hemligheter och konfiguration

- **Checka aldrig in lösenord, certifikat eller privata nycklar** i versionshantering. Använd `.gitignore` och granska historiken om en hemlighet råkat committas — den måste då betraktas som komprometterad och bytas ut.
- **Använd en hemlighetstjänst** (HashiCorp Vault, AWS Secrets Manager, Kubernetes Secrets med kryptering i vila m.fl.) i stället för miljövariabler i klartext för känsliga värden i produktion.
- **Rotera lösenord och certifikat** regelbundet och alltid omedelbart vid misstanke om kompromiss.

Behov av ytterligare stöd

Adda/SKR tillhandahåller administrativt stöd via klassa@skr.se

Har ni behov av djupare tekniskt stöd rekommenderar Adda/SKR att avropa konsultstöd inom ramen för ramavtal IT-konsulttjänster 2021 vid dynamisk rangordning inom kompetensområdet IT-, informations- och cybersäkerhet.

Se avropsvägledningen för ytterligare information:
https://klassa.skr.se/sidor/stodmaterial/a-klassa-5-vagledning