Some thoughts on INDIGO IAM, endpoints, static content and HAProxy

Mischa Sallé, Nikhef

INDIGO IAM Hackathon pre-day, CERN, 10 Feb 2025

INDIGO IAM / endpoints

- Endpoints relying on INDIGO IAM DB:

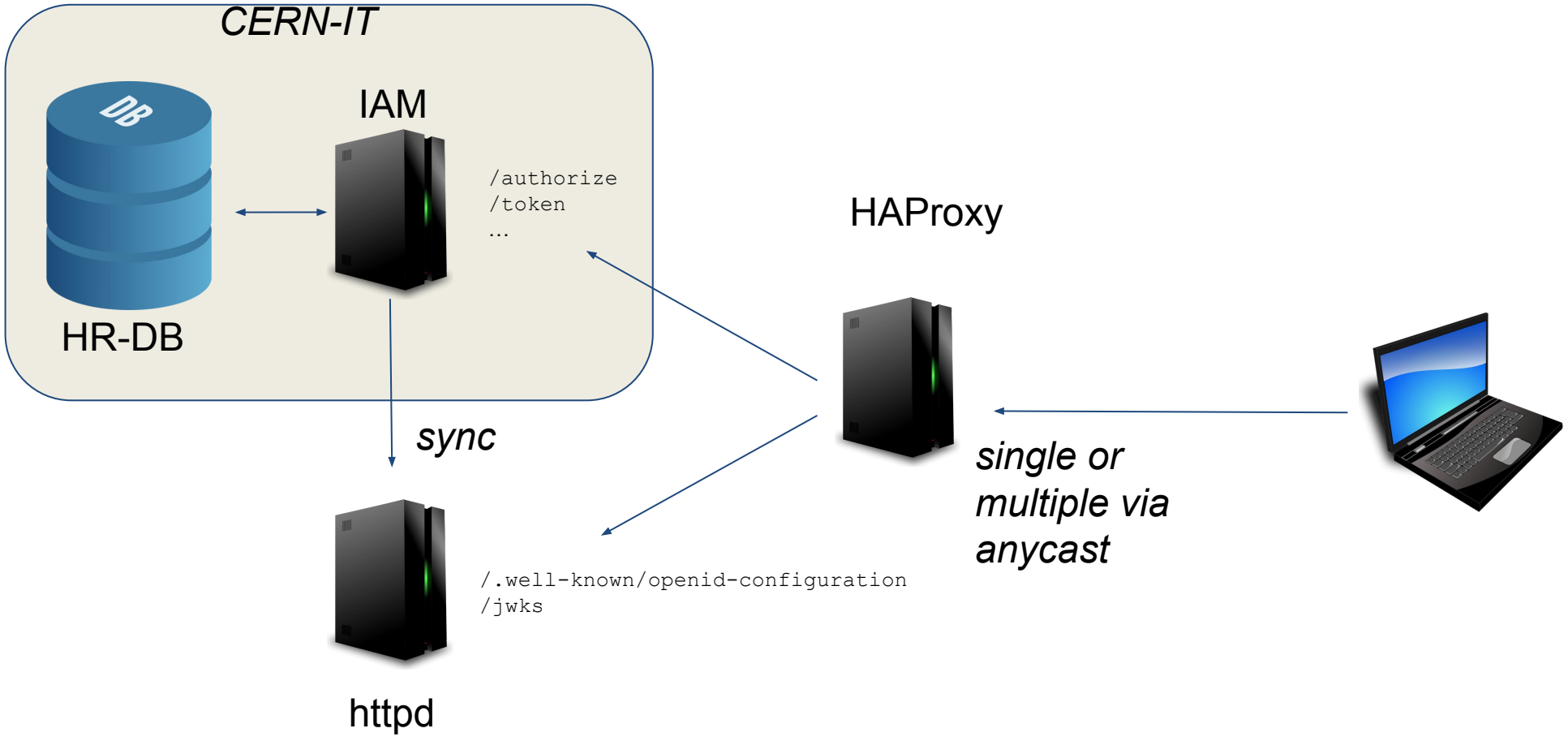
  `/token, /authorize, /introspect, ...`

- But: `/.well-known/...` and `jws_uri` (typically `/jwk`) don't
  - Only ones needed by RP (Relying Party) for *offline* validation:
    a. `iss` claim → `/.well-known/openid-configuration`
    b. `/.well-known` endpoint → `jwks_uri`
    c. `jwks_uri` → JWT signing key
  - Endpoints can be served semi-statically, just plain httpd
  - Can have the contents synchronized periodically

Splitting endpoints

- Two main options, either:
  - Run the plain httpd on the VO issuer hostname & INDIGO IAM elsewhere
    - `.well-known` takes care of other endpoints
  - Run a reverse proxy, e.g. HAProxy on the VO issuer hostname:
    - Redirect to INDIGO IAM or plain httpd depending on requested URL
- (Probably) no need for software changes, only puppet/ansible/… and extra VMs
  - INDIGO IAM must be able to config/override issuer
- HAProxy is robust and easy to set up, more flexibility (next slide), but extra VM

Example setup



Leveraging HAProxy

- HAProxy can be run by e.g. experiments
  - No reliance on CERN IT
- Can run multiple I-IAMs behind single HAProxy
  - Automatic fail-over (probably best active-passive, see next slide)
  - Easy for maintenance
- Suspension: can block JWT key in either HAProxy or httpd
- Can run multiple HAProxies and use anycast to keep single IP/hostname (used in production by RCauth online CA)

Extra slide

Extra: Multi IAM: Active-active / DB sync

- Since verification is offline using JWT keys:
  - Typically only `/authorize` and `/token` endpoints relying on DB
  - Still need session/database synchronization between these calls
  - Might be able to ensure both are accessed from same network