UAA Operator Issues

25 Feb 2013

uaa.yml

“things are the way they are because they got that way” – Gerald Weinberg

Let's walk through this file:

https://github.com/vmware-ac/portal2/blob/master/config/backend/uaa.yml

configuring secrets and keys

- configuring the uaa with public key: https://github.com/vmware-ac/deployments/blob/master/staging/staging.yml#L818
- avoiding the phantom sre user problem
- bootstrap clients and secrets
- bootstrap users and group membership
- override issues
- dev-template => manifest.yml => uaa.yml.erb => uaa.yml => spring *.xml => uaa

running the uaa locally

- mvn clean install; cd uaa/uaa; mvn tomcat7:run
- *or* mvnDebug tomcat7:run
- running a login server and uaa locally
  - from root with parallel checkouts of /uaa and /login-server
  - cd uaa
  - mvn clean install
  - cd ../login-server
  - mvn clean install
  - mvn tomcat:run -P integration

checking health with uaac

- targets and contexts
- info -- commit id, links
- varz
- be the client
- decode tokens
- be a user whoami, context

logs

- typical tomcat, spring logs
- some exceptions for unnecessary database migrations are logged and can be ignored
- making sense of it all

uaa db

- default is in memory
- can specify postgres, soon mysql

testing the uaa

- unit tests:
  - from uaa or uaa/uaa
  - mvn test
- integration tests:
  - from uaa with local login-server and uaa running
  - mvn integration-test
- uaac integration tests
  - `UAA_CLIENT_TARGET=http://localhost:8080/uaa bundle exec rake test`
- yeti
- uaa/gatling: see http://gatling.cloudfoundry.com/reports.html

login server

- another java app
- primarily provides the oauth2 /authorize endpoint services, but sometimes proxies other endpoints to the uaa
- contains any ui related to authentication and approvals
- uaa has small, generic login server built In, but the login_server job provides CF artwork and UI
- no db, just talks to the uaa
- debug and deploy much like the uaa