Project 2 - Deception component generator

Cybersecurity experts have recently proposed using defensive deception as a means to leverage the information asymmetry typically enjoyed by attackers as a tool for defenders. By creating fake services and components that appear as valuable targets to attackers, defenders can divert the attacker's attention and resources away from critical assets. Attackers might spend time and effort trying to compromise these fake elements, leaving less capacity to target actual valuable assets. The goal of this project is to create a fake resource generator for one specific type of resource.

Specification

- The final output of the generator must be a OCI compatible image ready to be instantiated
- The container image must include all the relevant "fake" data for its correct operation
- The container must function out of the box, eventual configuration has to be provided during the generation to build the final working image

You can choose one of the following resource types to implement

- Relational Database
  - auto generate complex schema and data from scratch
  - option to import an existing schema and generate data to populate it
  - configurable settings for client connections (port, authentication, credentials)
- LDAP server
  - auto generate users, groups, directory schema from scratch
  - option to import an existing schema and generate data to populate it
  - configurable settings for client connections (port, authentication, credentials)
- openID/oauth2.0 provider (eg. https://www.keycloak.org/)
  - configurable settings at generation time
  - automatic populate the database with fictional users (generated or imported list)
  - configure security groups and oath flow to secure api endpoints(eg. https://stackoverflow.blog/2022/12/22/the-complete-guide-to-protecting-your-apis-with-oauth2/)

- file server (SAMBA)
  - support both public and private shares
  - automatically generated and realistic fs hierarchy
  - populated with generated word/pdf files (https://pandoc.org/)
  - support for ldap for authentication
- rest API server
  - form an openAPI specification implement a dumb service with fake/random data
  - accept configurations for oauth authentication (at generation time) to secure the api endpoint and provide an out of the box functionality

References

You can use LLM to generate the data, but it is better to use a local first model instead of relying on web api like chatGPT.

- https://python.langchain.com/docs/integrations/llms/llamacpp
- https://medium.com/@karankakwani/build-and-run-llama2-llm-locally-a3b393c1570e