# Security Store Service for MongoDB Microservice

## Current Security Store Service Architecture

* PKI-setup to create self-signed CA as well as certificates for API-Gateway service
* Edgex-vault-worker to initialize the vault environment and service for API-gateway service
* API-Gateway service is consuming security store service on REST API endpoint v1/secret/edgex/pki/tls/edgex-kong
* Implementation same between Docker/snaps

## MongoDB Microservice and Related Credentials

* Docker-edgex-mongo repo
* Implemented with JavaScript
* Credentials required to initialize mongoDB
	+ Initial admin/password to create mongoDB instance
	+ Each individual username/password pair for other microservices in EdgeX
		- Metadata
		- Coredata
		- Rules\_engine
		- Notification
		- Scheduler
		- logging
* Shell script to retrieve access token and credentials

## Proposed Feature Added into Security Store for MongoDBInit Microservice

* path to consume the credentials v1/secret/edgex/mongoDBInit
* structure to be retrieved

{

“auth”: null,

“data”: {

“mongousername”: “user”,

“mongopasswd”: “passwd”,

“metadatabaseusername”: “metadata”

“metadatabasepassword”: “password”

“guid”: ”xxxxxxxxx-xxxxxxxx-xxxxxxxx”

},

* location of new featured added: edgex-vault-worker
* set up initial credentials for mongoDB in edgex-vault-worker
	+ Configuration file/hardcoded
	+ GUID generator
	+ Vault plug-in for credential generator
	+ Built-in database secret engine for mongoDB credential generator

## Additional Features/Enhancement for Security Store Service

* Option to replace self-signed CA with 3rd-party CA certificate
* Token associated with different microservices
* Policy (HCL/JSON) for tokens
* Access Token protection