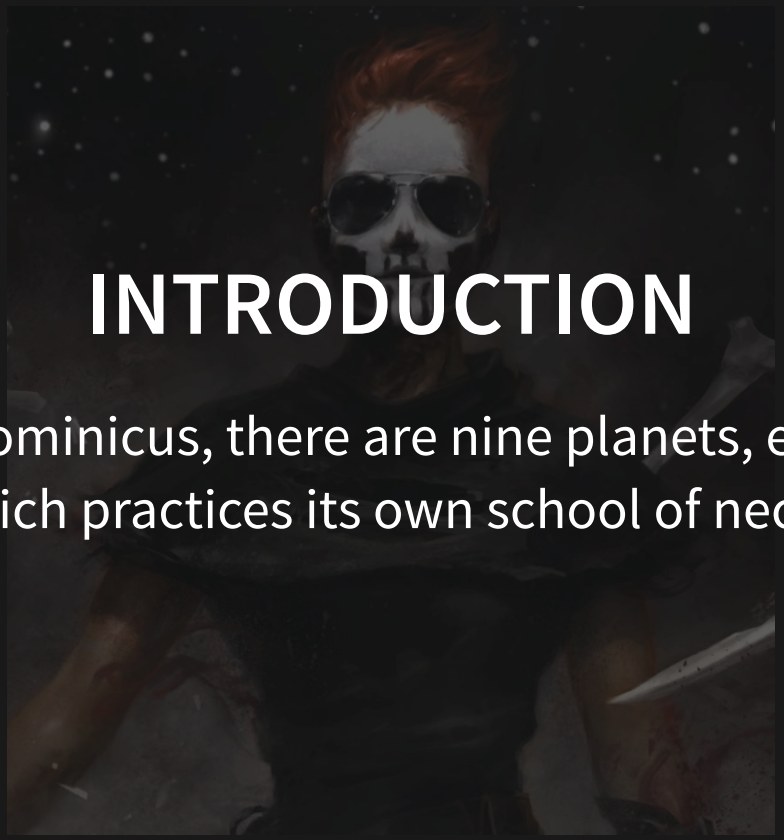
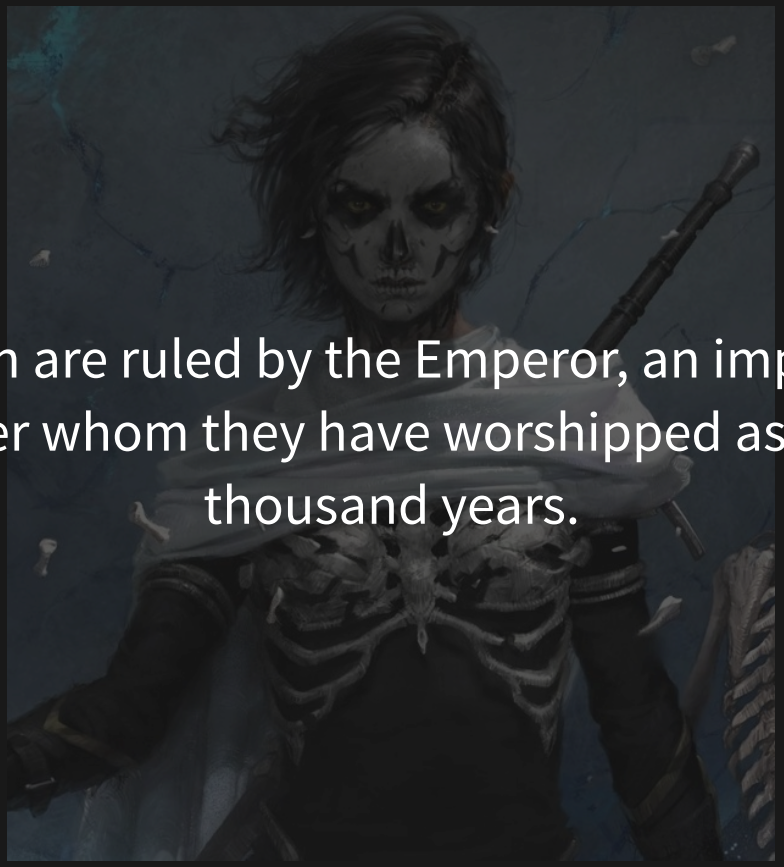

THE SHOW:
THE LOCKED TOMB:
CONTAINERS, SERVERS AND CERTIFICATES
maximba



INTRODUCTION

In the star system Dominicus, there are nine planets, each home to a great House which practices its own school of necromancy.

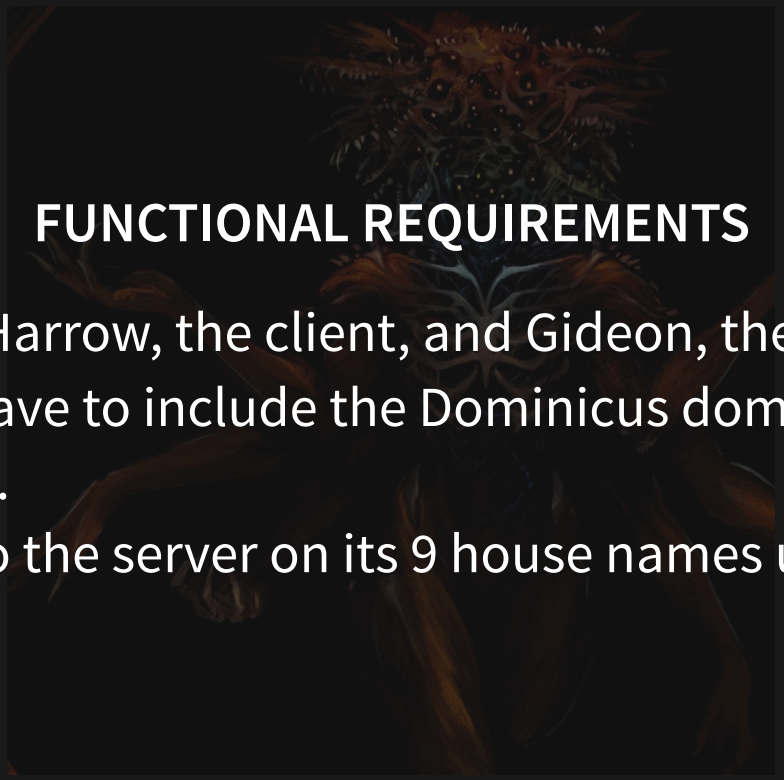


The Houses in turn are ruled by the Emperor, an impossibly powerful, immortal necromancer whom they have worshipped as a god for the past ten thousand years.

TL;DR

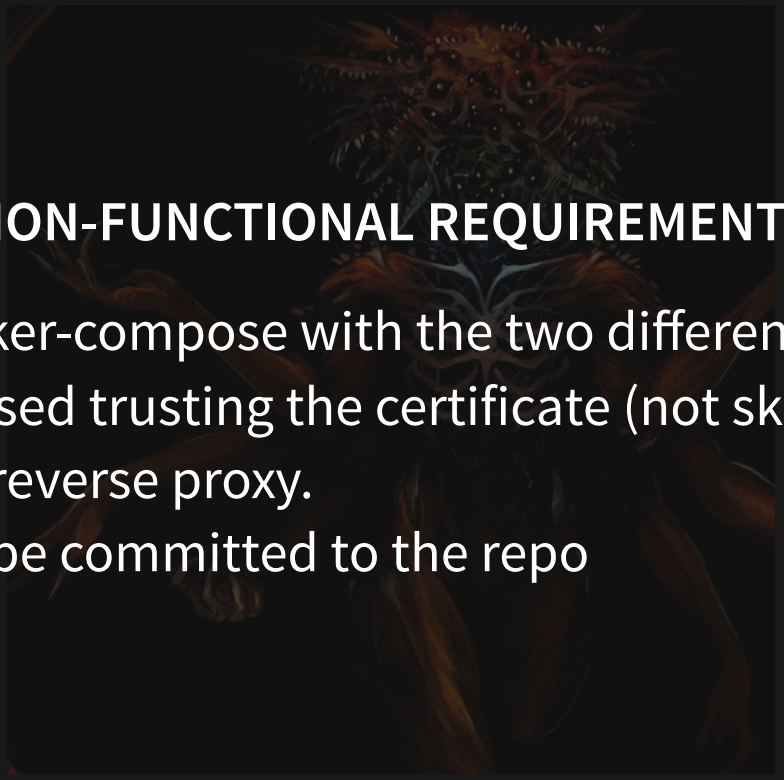
Be a Lyctor !

Lyctors are immortal necromancers, revered as saints, who serve as the Emperor's right-hand necromancers in wars against his enemies.



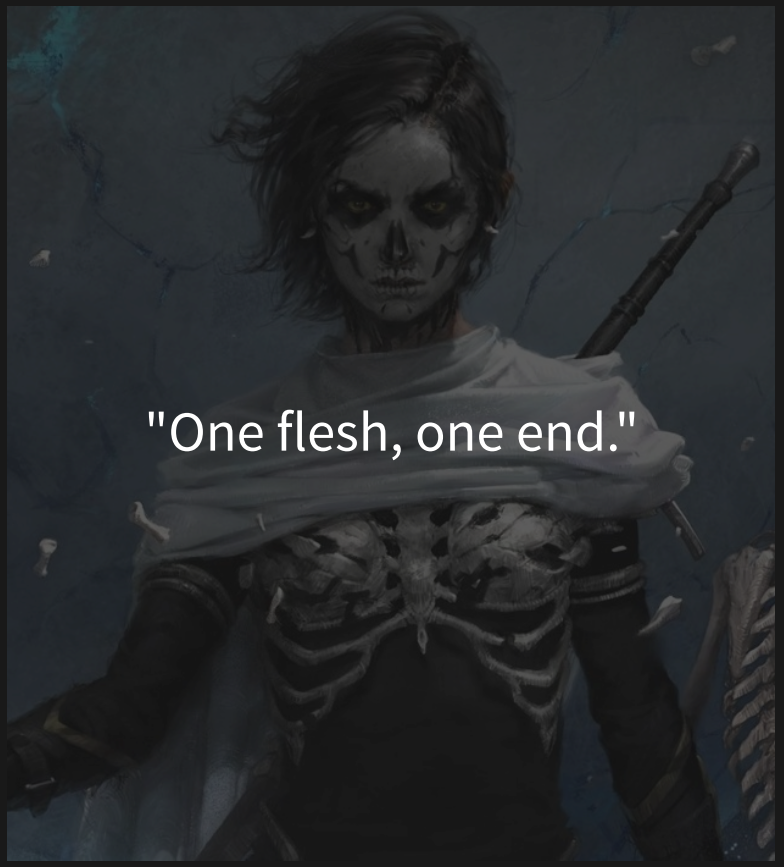
FUNCTIONAL REQUIREMENTS

- Build two services: Harrow, the client, and Gideon, the server
- Gideon responses have to include the Dominicus domain and the house ip. Like an echo service.
- Harrow has to call to the server on its 9 house names using DNS resolution



NON-FUNCTIONAL REQUIREMENTS

- Generate a docker-compose with the two different services
- HTTPS will be used trusting the certificate (not skipping validation).
- Do not use any reverse proxy.
- Secrets cannot be committed to the repo


"One flesh, one end."

FOLDER ELEMENTS

```
.
└── challenge
    ├── harrow
    │   ├── harrow.py
    │   ├── Dockerfile
    │   ├── Makefile
    │   └── requirements.txt
    ├── compose.yaml
    ├── LICENSE.txt
    ├── Makefile
    ├── minica
    │   ├── go.mod
    │   └── main.go
    ├── README.md
    └── gideon
        ├── Dockerfile
        ├── Makefile
        ├── requirements.txt
        └── gideon.py
```

Makefile

File: **Makefile**

```
# Var Definition
#
DOMAIN=${MX_domain}
CERT_DIR=${MX_certspath}
CERT_GEN=bin/minica

.PHONY: help certs build push all

help:
        @echo "Makefile arguments:"
        @echo ""
        @echo "pyvers - Python Image Base Version, default is: 3.11-slim-bullseye"
        @echo ""
        @echo "Makefile commands:"
        @echo "certs"
        @echo "build"
        @echo "push"
        @echo "all"

.DEFAULT_GOAL := all

certs:
    @echo "Building Cert generator"
    @([ ! -d bin ] && mkdir -p bin) || true
    @([ ! -f $(CERT_GEN) ] && (cd minica; go build -o ../bin) ) || true
    @echo "Generating Certs"
    @([ ! -d $(CERT_DIR) ] && mkdir -p $(CERT_DIR)) || true
    @cd $(CERT_DIR) && ../$(CERT_GEN) \
        -ca-cert root-ca.pem \
        -ca-key key-ca.pem \
        -domains '$(DOMAIN),*.$(DOMAIN),localhost' \
        -ip-addresses 127.0.0.1 || true
build:
    $(MAKE) -C gideon build
    $(MAKE) -C harrow build

push:
    $(MAKE) -C gideon push
    $(MAKE) -C harrow push

all: certs build
```

gideon.py

```python
if __name__ == "__main__":
    domain, ca_root, key, cert = init_vars()
    https = MxGideon(key, cert, HttpsHandler, bind_port=8443, is_tls=True)
    http = MxGideon(key, cert, HttpHandler, bind_port=8080, is_tls=False, caroot=ca_root)
    print(f"Gideon started at https://{https.bind_address}:{https.bind_port}")
    print(f"Gideon started at http://{http.bind_address}:{http.bind_port}")

    try:
        run(http, https)
    except KeyboardInterrupt:
        pass

    https.server_close()
    http.server_close()
    print("Gideon stopped.")
```

harrow.py

```python
def client(session, domain, ca_file):
    try:
        session.get(f'https://{domain}:8443/')
    except (requests.exceptions.SSLError, OSError):
        response = requests.get(f'http://{domain}:8080/root_ca')
        if response.ok:
            custom_ca = response.content
            with open(ca_file, 'wb') as outfile:
                outfile.write(custom_ca)
    try:
        response = session.get(f'https://{random_server(domain)}:8443/')
    except requests.exceptions.SSLError as err:
        print(f'SSL Error. {err}')
    else:
        print(response.text)


def run(session, domain, ca):
    while True:
        client(session, domain, ca)
        time.sleep(3)


if __name__ == "__main__":
    dom, caroot = init_vars()
    sess = create_session(caroot)
    try:
        run(sess, dom, caroot)
    except KeyboardInterrupt:
        sess.close()
        print("Harrow Stopped")
```

docker-compose.yaml - server

```yaml
server:
  hostname: gideon
  build:
    context: ./gideon
    args:
      pyvers: 3.11-slim-bullseye
  image: maximba/gideon:1.0.0
  secrets:
    - cert.pem
    - key.pem
    - root-ca.pem
  environment:
    - MX_domain=${MX_domain}
  ports:
    - '443:8443'
    - '80:8080'
  command: ["python","gideon.py"]
```

docker-compose.yaml - client

```yaml
client:
  hostname: harrow
  links:
    - "server:${MX_domain}"
    - "server:house1.${MX_domain}"
    - "server:house2.${MX_domain}"
    - "server:house3.${MX_domain}"
    - "server:house4.${MX_domain}"
    - "server:house5.${MX_domain}"
    - "server:house6.${MX_domain}"
    - "server:house7.${MX_domain}"
    - "server:house8.${MX_domain}"
    - "server:house9.${MX_domain}"

  build:
    context: ./harrow
    args:
      pyvers: 3.11-slim-bullseye
  image: maximba/harrow:1.0.0
  environment:
    - MX_domain=${MX_domain}
  command: ["python","harrow.py"]
  depends_on:
    - server
```

SPECIAL THANKS TO:

- *Minica*, the Certificates Generator at https://github.com/jsha/minica
- *The Locked Tomb Wiki* at https://thelockedtomb.fandom.com/wiki/The_Locked_Tomb_Wiki
- *Tamsyn Muir*, the author of the Locked Tomb books

THANK YOU!

DON'T FORGET TO VISIT !

- Code repo at https://github.com/maximba/challenge
- This presentation at https://github.com/maximba/slides

maxi@bxsoft.com