CTF systém v Kubernetes

Jan Stránský

CTF

- Capture The Flag
- Soutěž v IT bezpečnosti
- Úkoly z různých oblastí
- Moderní styl vzdělávání

Proč a motivace

- Naučení se pracovat s mikroslužbami
- Lepší pochopení Kubernetes API
- Pokus o on-prem řešení typicky cloudového problému
- Inspirace
  - CTFd
  - Soutěže Haxagon Skirmish a Kybersoutež
- Zájem o CTF

Figure: Kubernetes logo

Požadavky projektu

- Webové rozhraní
- Úlohy spustitelné na Kubernetes
- Komunikace přes webshell
- REST API
- Spustitelné v prostředí školního Kubernetes clusteru
- Rozšiřitelnost
- Stateless

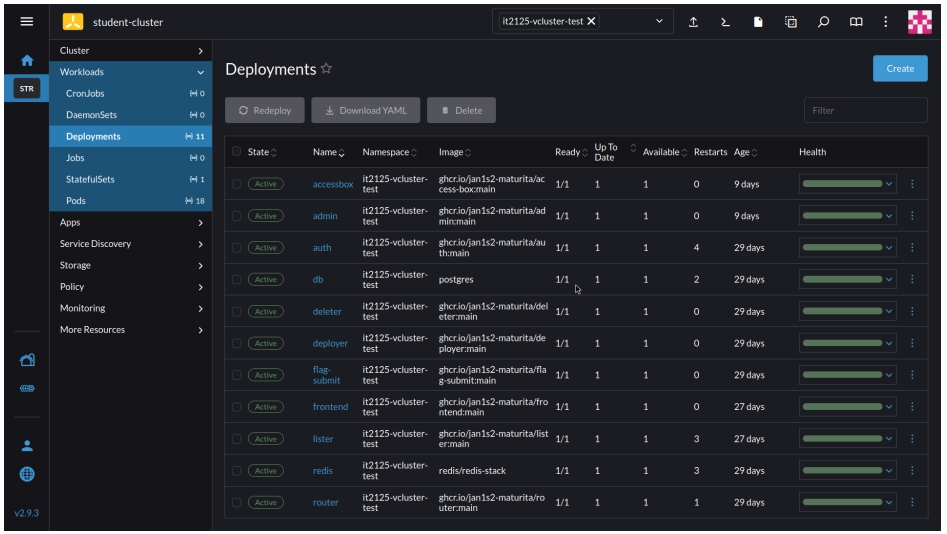

Figure: Deployments in Kubernetes

Technologie

- React
- Kubernetes
- FastAPI
- SQLAlchemy
- JWT
- Redis
- Mikroslužby

Databáze

- Hlavní = PostgreSQL
- Pro aktuálně spuštěné úkoly = Redis

```
E  1 from .db import User, Challenge, Image, Flag, Service
E  2 from sqlalchemy.orm import sessionmaker      ■ Import sqlalchemy.orm se nepovedlo vyřešit
E  3 from sqlalchemy import create_engine      ■ Import sqlalchemy se nepovedlo vyřešit
   4 # database connection class - postgres
   5 class Database:
   6     def __init__(self, host, port, user, password, db_name):
   7         self.engine = create_engine(f'postgresql://{user}:{password}@{host}:{port}/{db_name}')
   8         self.Session = sessionmaker(bind=self.engine)
   9         self.init_db()
  10     def init_db(self):
  11         Flag.metadata.create_all(self.engine)
  12         User.metadata.create_all(self.engine)
  13         Challenge.metadata.create_all(self.engine)
  14         Image.metadata.create_all(self.engine)
  15         Service.metadata.create_all(self.engine)
  16
  17     def get_session(self):
  18         return self.Session()
  19     def get_user_by_id(self, user_id):
  20         session = self.get_session()
  21         user = session.query(User).filter_by(id=user_id).first()
  22         session.close()
  23         return user
```

Figure: Database connection

Problémy

- Logy z Access boxu
- Cookies
- Routing mezi frontendem a backendem pomocí Ingress
- Komunikace s Kubernetes

```
def execute_command(self, user_id, command):
    # command = ["sh", "-c", f"'{command} >> /tmp/output.txt 2>&
    # command_new = ["sh" "-c", f"'{command} >> /tmp/output.txt
    command_new = [
            "sh",
            "-c",
            f"{command} >> /tmp/output.txt 2>&1"
            ]

    try:
        # self.v1.connect_get_namespaced_pod_exec("accessbox", s
        stream(self.v1.connect_get_namespaced_pod_exec,
               "accessbox",
               self.get_user_namespace(user_id),
               command=command_new,
               stderr=True,
               stdin=False,
               stdout=True,
               tty=False)
        return True
    except Exception as e:     ■ e není přístupné
        return False
```

Figure: Řešení problému s logy

Backend služby

- Router
- Auth (klíč, PostgreSQL)
- Admin (k8s, PostgreSQL)
- Deployer (k8s, Redis, PostgreSQL)
- Lister (Redis, PostgreSQL)
- Deleter (k8s, Redis)
- Flag submit (PostgreSQL)

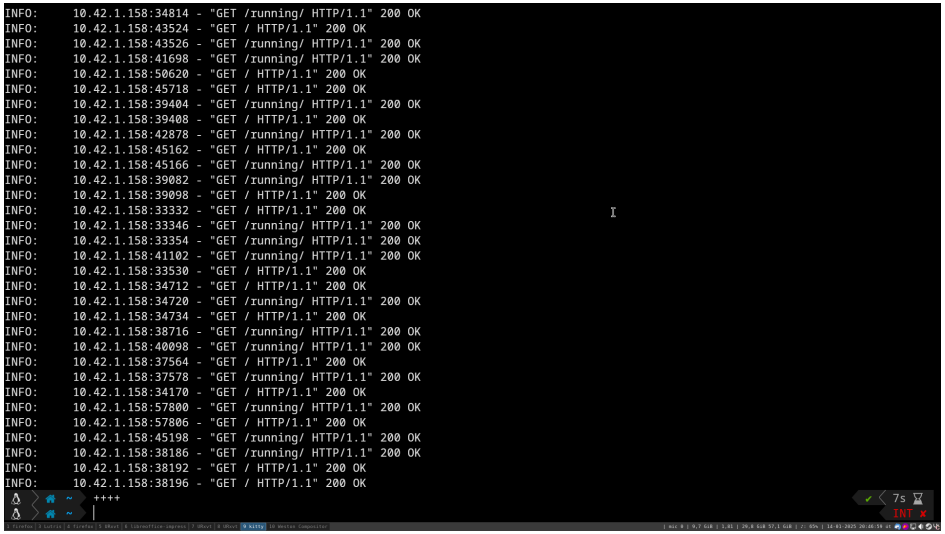

Figure: Logy listeru

Access box

- Porovnání s VPN nebo přímým přístupem
- Kali Linux kontejner
- Webový přístup
- Přístup k úkolům

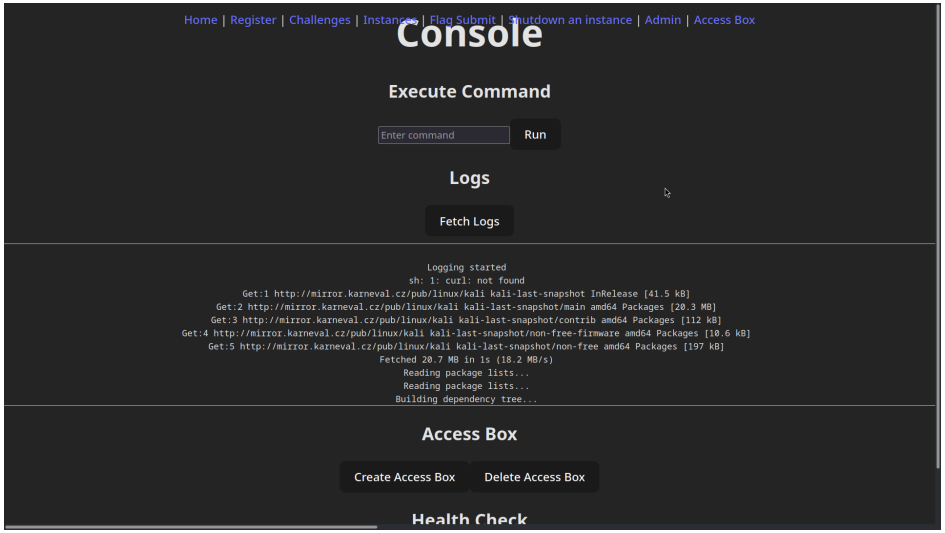

Figure: Access box

Nesplněné cíle a budoucí rozšíření

- Dynamické skóre
- Leaderboard
- Seznam již vyřešených úkolů
- Zobrazení bodů
- Vylepšené chybové hlášky
- Bezpečnostně nepředpokládat oddělený cluster pro úkoly
- Ověřování Kubernetes certifikátu

Děkuji za pozornost