Python Side-Cars

Moshe Zadka – https://cobordism.com

Acknowledgement of Country

Belmont (in San Francisco Bay Area Peninsula)
Ancestral homeland of the Ramaytush Ohlone

Quick Kubernetes Refresher

Let's get on the same page!

Quick Kubernetes Refresher

Let's get on the same page!
Not

Quick Kubernetes Refresher

Let's get on the same page!
Not comprehensive

Quick Kubernetes Refresher

Let's get on the same page!
Not comprehensive Kubernetes

Quick Kubernetes Refresher

Let's get on the same page!
Not comprehensive Kubernetes review!

Quick Kubernetes Refresher

Let's get on the same page!
Not comprehensive Kubernetes review!

Kubernetes Refresher: Pods and Containers

- Pods: basic execution unit
- n Containers: 1 Pod

Kubernetes Refresher: Pods and Containers

- Pods: basic execution unit
- n Containers: 1 Pod
- Share: network
- Maybe share: process
- Maybe share: volume mounts
- Never share: file systems

Kubernetes Refresher: Pod Readiness

- Answers "is the Pod good"?
- Most common: TCP/HTTP checks
- Sometimes: command checks

Kubernetes Refresher: Deployments and Services

- Deployment: ReplicaSet + Service
- ReplicaSet: collection of identity-less Pods
- Service: Routing to "good" Pods

Side-Car Pattern

- Extra container in Pod
- Main container "does the thing"
- Side-car "takes care of the rest"

Kubernetes Refresher: Side-Car Example

- Main container runs a web application caching to files
- Side car container garbage-collects files

Why Side-Cars?

- Separate resource limits
- Legible to K8s dashboards

Why Side-Cars?

- Separate resource limits
- Legible to K8s dashboards
- *Simplifies containers*

Side-Car: Readiness

- Check file existence/time-stamp
- Check contents of HTTP response
- Arbitrary logic
- Python better than shell-in-YAML

Side-Car: Metrics

- Reformat into Prometheus
- Synthetic checks
- Scan a (log) file

Side-Car: Scheduled Tasks

- Reindexing
- Garbage collecting

Why Python Side-Cars?

What makes Python a good fit?

Prototype Side-Car

Side car need often realized late in development process

Prototype Side-Car

Side car need often realized late in development process
Fast prototyping rewarded!

Iterate on Side-Car

How does it look in the dashboard?

Iterate on Side-Car

How does it look in the dashboard?
Fast iteration speed rewarded!

Adapt Side-Car

Container changed? Side car needs to be adapted!

Adapt Side-Car

Container changed? Side car needs to be adapted!
Fast modification speed when chasing container code rewarded!

Examples of Side-cars

Challenge: Fit side car on a slide

Examples of Side-cars

Challenge: Fit side car on a slide
Note: Code optimized for slides,

Examples of Side-cars

Challenge: Fit side car on a slide
Note: Code optimized for slides, not best practices!

Readiness side-car

```
def check_readiness(request):
    result = httpx.get(
        "http://127.0.0.1:8080/known-value"
    )
    if result.json()["value"] != 5:
        1/0
    return Response("ok")

with Configurator() as config:
    config.add_route('check', '/')
    config.add_view(
        check_readiness,
        route_name='check',
    )
    application = config.make_wsgi_app()
```

Metrics side-car

```
registry = CollectorRegistry()
latency = Gauge('latency', registry=registry)
def metrics(request):
    before = time.time()
    result = httpx.get("http://127.0.0.1:8080")
    after = time.time()
    latency.set(after - before)
    return Response(
        generate_latest(registry),
        content_type=CONTENT_TYPE_LATEST,
    )
with Configurator() as config:
    config.add_route('metrics', '/metrics')
    config.add_view(
        check_readiness,
        route_name='metrics',
    )
    application = config.make_wsgi_app()
```

Scheduler side-car

```
while True:
    httpx.get("http://127.0.0.1:8080/flush_queues")
    time.sleep(60)
```

Advanced example: Log analyzer

- Main container logs to /var/log/something
- Side-car processes log

Log analyzer: The Pod

```
...
kind: Pod
spec:
  shareProcessNamespace: true
...
```

Finding Our Roots

```
proc = pathlib.Path("/proc")
my_root_inode = (
    proc /
    str(os.getpid()) /
    "root"
).stat().st_ino
```

Finding Our Peer's Roots

```
for process in proc.glob("[1-9]*"):
    inode = (
        process /
        "root"
    ).stat().st_ino
    if inode != my_root_inode:
        break
peer_root = process / "root"
```

Analyzing log

```
log = peer_root / "var" / "log" / "something"
while True:
    with open(log) as fpin:
        lines = sum(1 for line in fpin)
        httpx.post(
            "http://metrics.example.com",
            json=dict(lines=lines)
        )
    time.sleep(60)
```

Final thoughts: Containers

- Black boxes
- Simple

Final thoughts: Container extension

- Pods allow adding from the outside
- Clear(-ish) separation of responsibilities

Final thoughts: Container API

- Containers can become more side-car friendly!
- Good APIs

Final thoughts: Write Side-Cars

- Packaging as containers? Good!
- Packaging as separate containers? Even better!