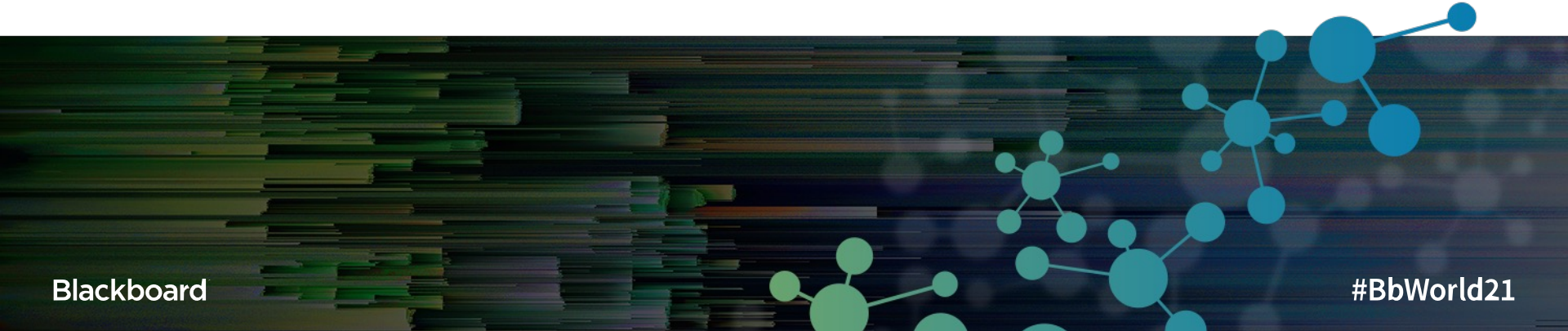
BbWORLD 21

Hosting REST and LTI integrations

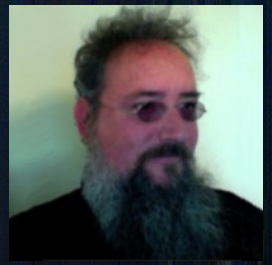
Using Container and Container Orchestration for the delivery of your REST or LTI applications.



Forward-Looking Statement

Statements regarding our product development initiatives, including new products and future product upgrades, updates or enhancements represent our current intentions, but may be modified, delayed or abandoned without prior notice and there is no assurance that such offering, upgrades, updates or functionality will become available unless and until they have been made generally available to our customers.

About Me…



Mark O'Neil

Director, Product Management, Platform and APIs
mark.oneil@blackboard.com

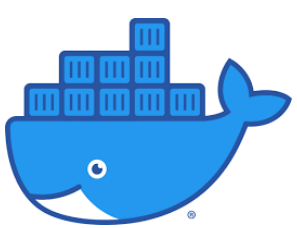
Work History:

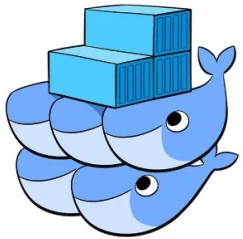
- 1982-93: Undergraduate Studies (BFA): Art History, Printmaking, Computer graphics, and Philosophy.
- 1993-1997: Director Scientific Illustration Purdue University
- 1997-2009: Curricular Systems/Software Engineer Dartmouth
- 2009-now: Blackboard

Computing Interests:

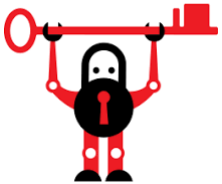
- APIs, AI/ML, FOSS, DevOps, Docker, PaaS/SaaS and Software/System Architecture

What is the technology behind this talk?





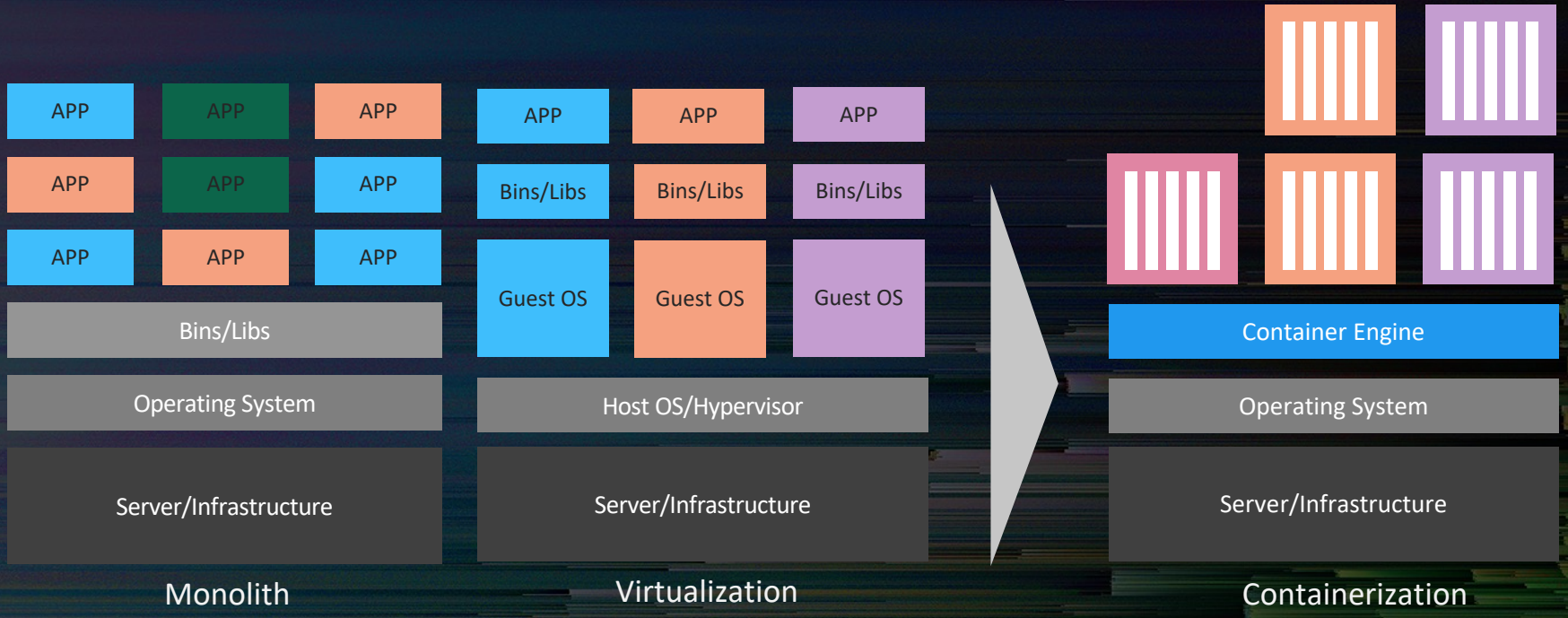
Containerization and Orchestration



SSL/Certificates/Dev env SSL

Routing and Management

What is Containerization?

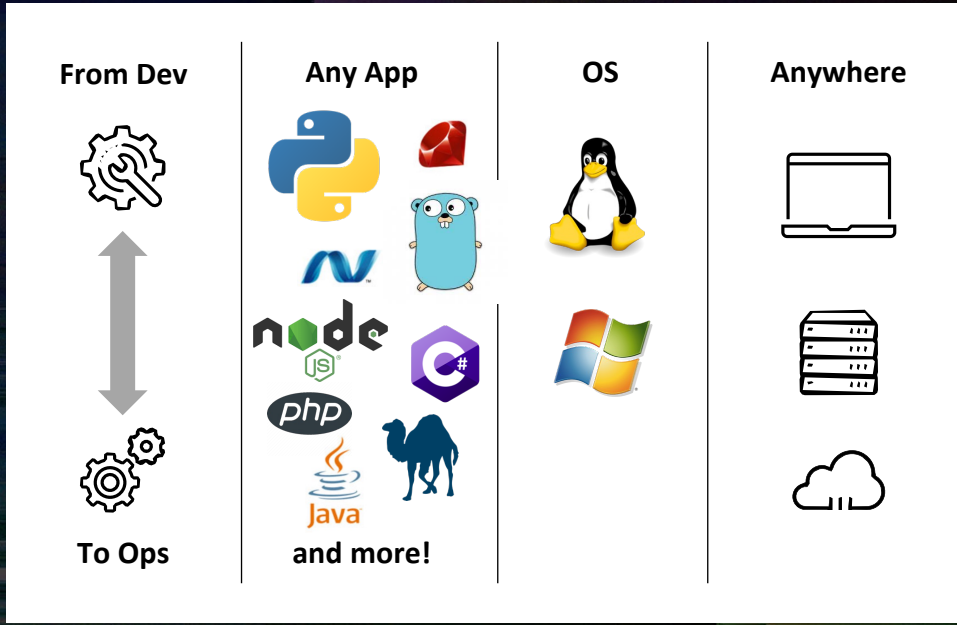


Why Use Containerization?

Containerization enables you to run applications in consistent environments defined by code which are deployable anywhere – desktop, machine room metal, cloud (PaaS)

They may be designed and configured for simple to complex use cases

They start fast* and are easily manageable for High Availability (Failsafe runtime, rolling restarts/updates)



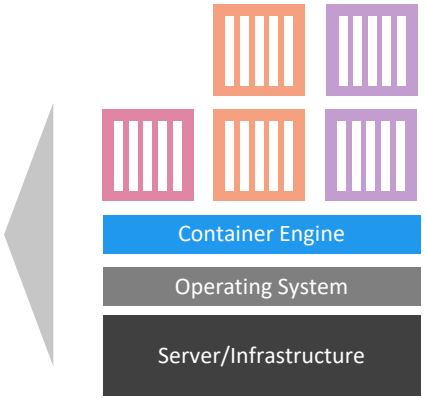
*dependent on how the container is built

https://github.com/moneil/dockerdemo-nginx-certbot
https://github.com/moneil/appN

In Summary Containerization benefits you because…

Containers offer environment reproducibility, isolation and security, consistent environments for teams, production delivery and super fast startup time over other software delivery models.

Consistency across environments
Infrastructure as code
Ease of deployment
Ease of management



Some Docker Basics

1. **docker build** – to build images from dockerfiles

   $ docker build –t <imagename[:version]> .

2. **docker run** – to start containers from built images (not my favorite way to start containers)

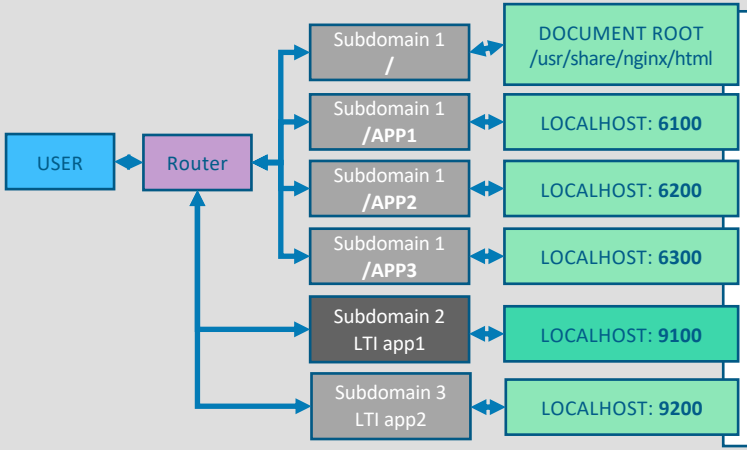
   $ docker run –d /
   [–p inboundport:containerport] /
   [–name runtimename] /
   <imagename>

3. **docker compose** – start one or several containers with consistency! Stop with same file.

   $ docker-compose up –d
   $ docker-compose down

Understanding Reverse Proxy

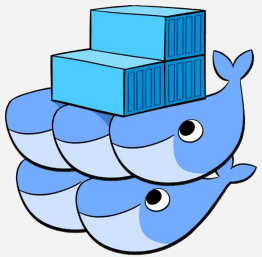
1. SSL/NGINX or Traefik ready server
2. Configure routing to proxy to application ports per URL
3. **Best Practice**: Separate REST Applications by *path*
4. ***LTI requires one app per subdomain!***



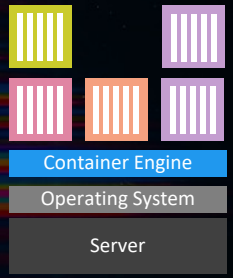
/etc/nginx/sites-available/lti-game.northbynortheast.io

```
server {
  server_name lti-game.northbynortheast.io;
  listen 443 ssl http2 ;
  #SSL settings…
  #location reverse-proxy
   location / {
       proxy_set_header X-Real-IP $remote_addr;
       proxy_set_header X-Forwarded-For  \
          $proxy_add_x_forwarded_for;
       proxy_set_header Host $http_host;
       proxy_set_header X-NginX-Proxy true;
       proxy_pass http://LTIGAME:8080;
       proxy_redirect off;
   }
}
```

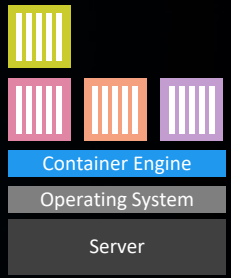
What is Container Orchestration?



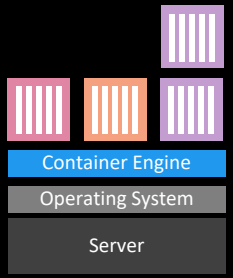
https://www.g2.com/categories/container-orchestration



Node 1



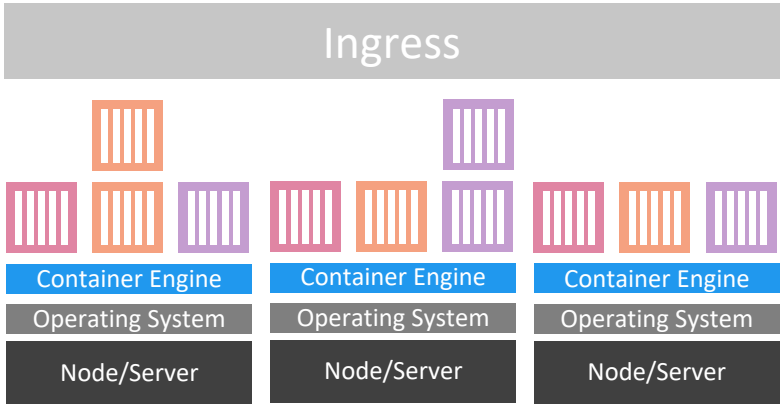
Node 2



Node 3 . . .

Why Container Orchestration?

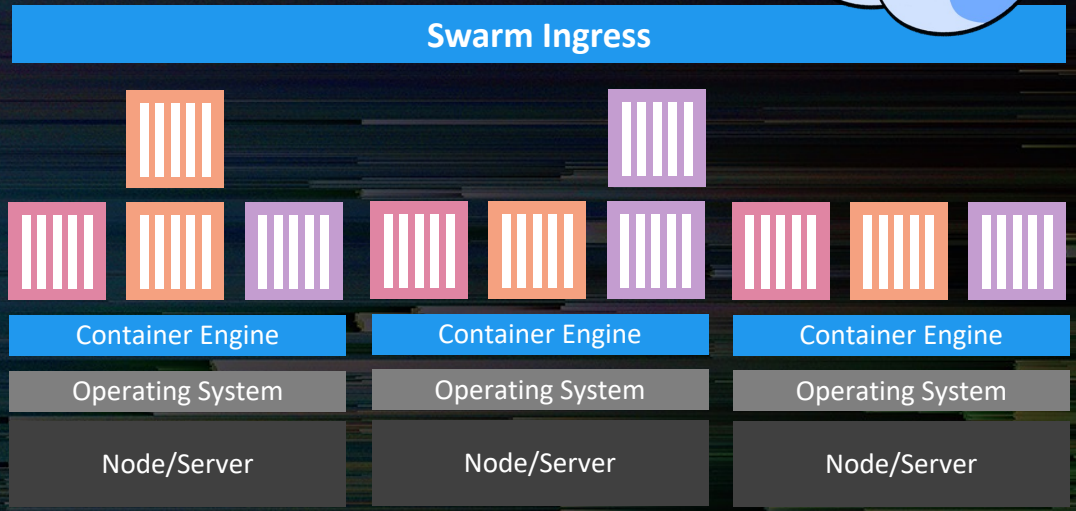
Container orchestration automates the multiple tasks necessary for robust service delivery at scale. Create services, tasks, and containers, auto restarts on application failure and rolling update management.



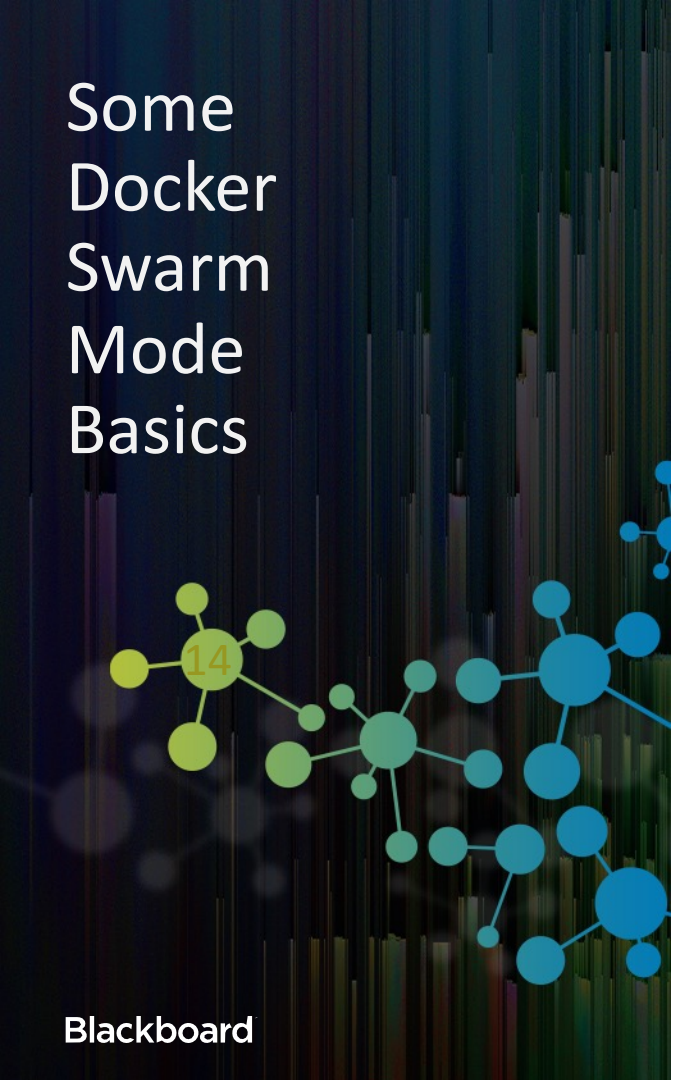
Docker Swarm (Specifically Docker Swarm Mode!)

Docker Swarm Mode is super easy to learn and while less capable of complex architectures than other orchestration software, it is more than capable of production at scale.

Every swarm requires a minimum of three nodes and one manager (odd number required to support manager node failures to maintain quorum)



Some Docker Swarm Mode Basics



Blackboard

Create a basic single manager, multi-worker swarm…

1. Initialize your swarm on your *first manager node*

   $ docker swarm init
   Swarm initialized: current node (4dgxs9kv4ewemlzg8qb2xzyi4) is now a manager.

2. Add a worker to this swarm, run the following command:

   $ docker swarm join --token
   SWMTKN-1- /
   0kf554rw9lya460bp77wd1s711ouayxhc/
   72bcdh1k8l9ppqq9g-9lya460bp77wd1s711z1bnm
   172.30.3.51:2377

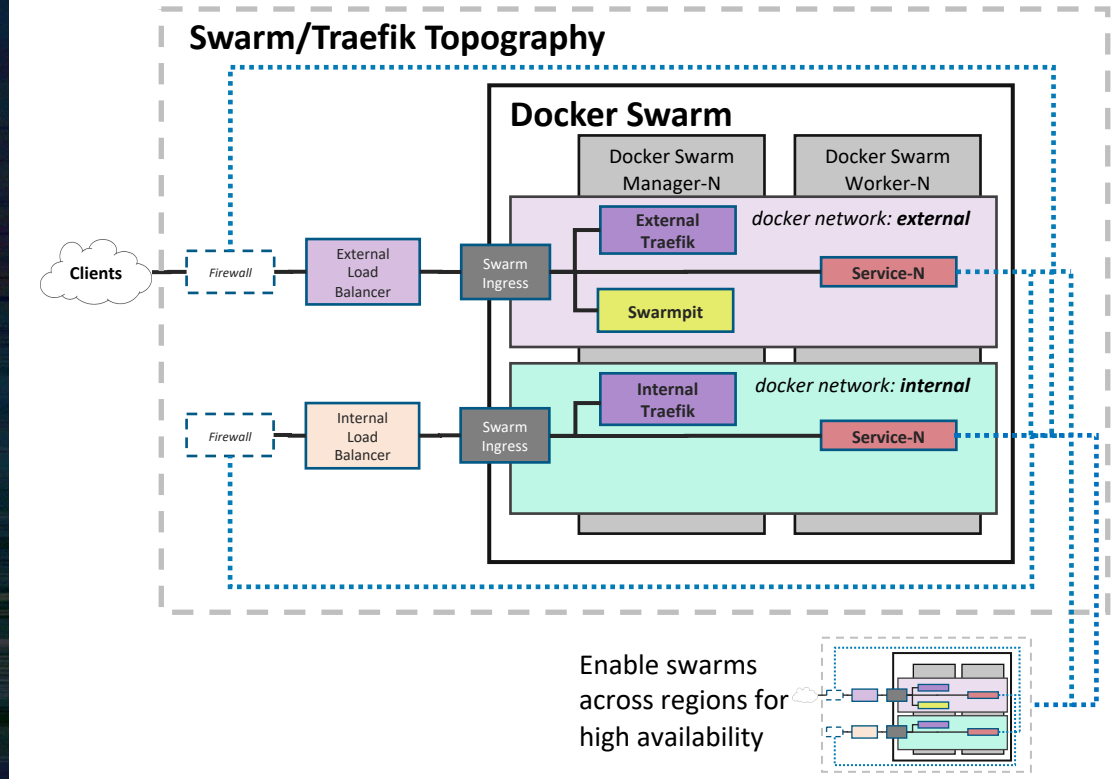
3. Add a manager to this swarm, run $ docker swarm join-token manager on a new node and follow the instructions.

4. Now SSH into each of your designated worker nodes and run the worker join command presented when you created your manager.

Swarm Mode Topography

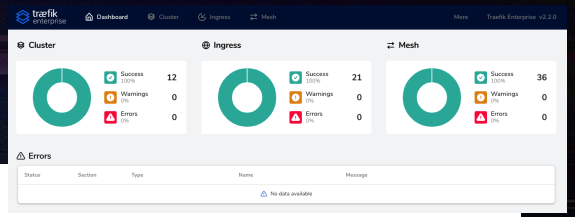
Key Topography concepts:

- 1-N Servers (PaaS or metal)
- Odd number of Manager nodes for quarum. Recommended no more than 7 managers
- NGINX or Traefik provide service routing
- SSL may be terminated at routing service to save on certificate costs
- More servers across zones/regions = more isolation, redundancy, availability
- Routers proxy to application ports per URL location directives
- Keep sensitive services on internal network!
- Best Practice: Separate RESTful Applications by *path*
- *IMPORTANT: LTI requires one app per subdomain!*



Traefik for Service reverse-proxy

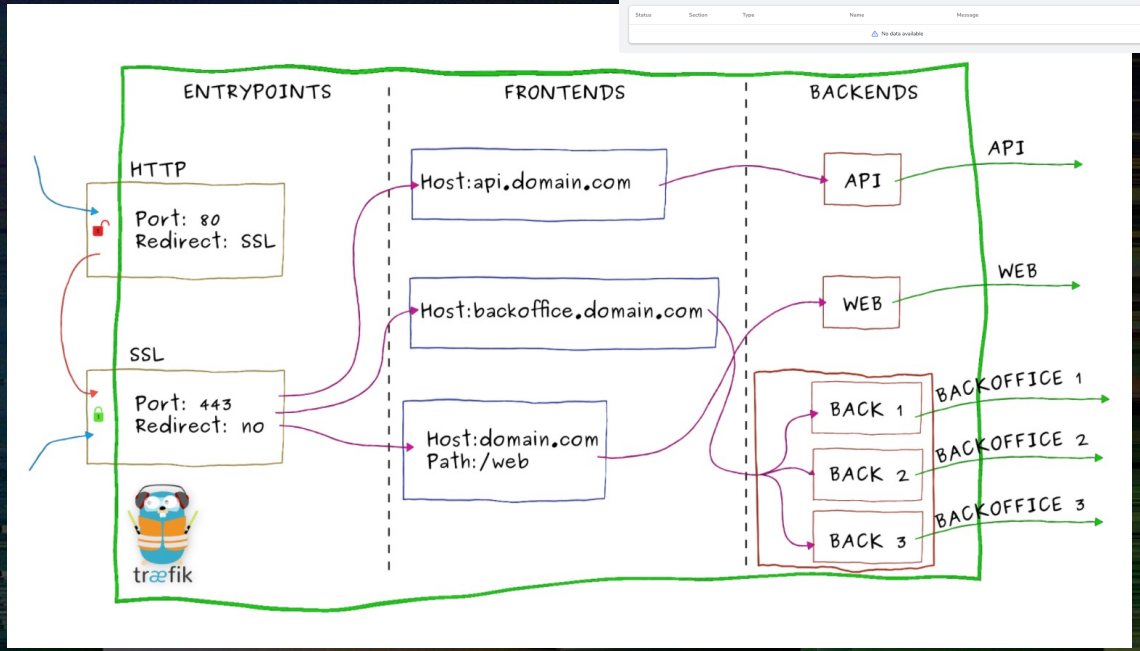
https://traefik.io and https://dockerswarm.rocks/traefik/



Easy installation
Provides:

- Container routing
- Inspection panel
- Auto cert management
- Load balancing

Requires minimal Traefik specific docker-compose.yml editing to add services

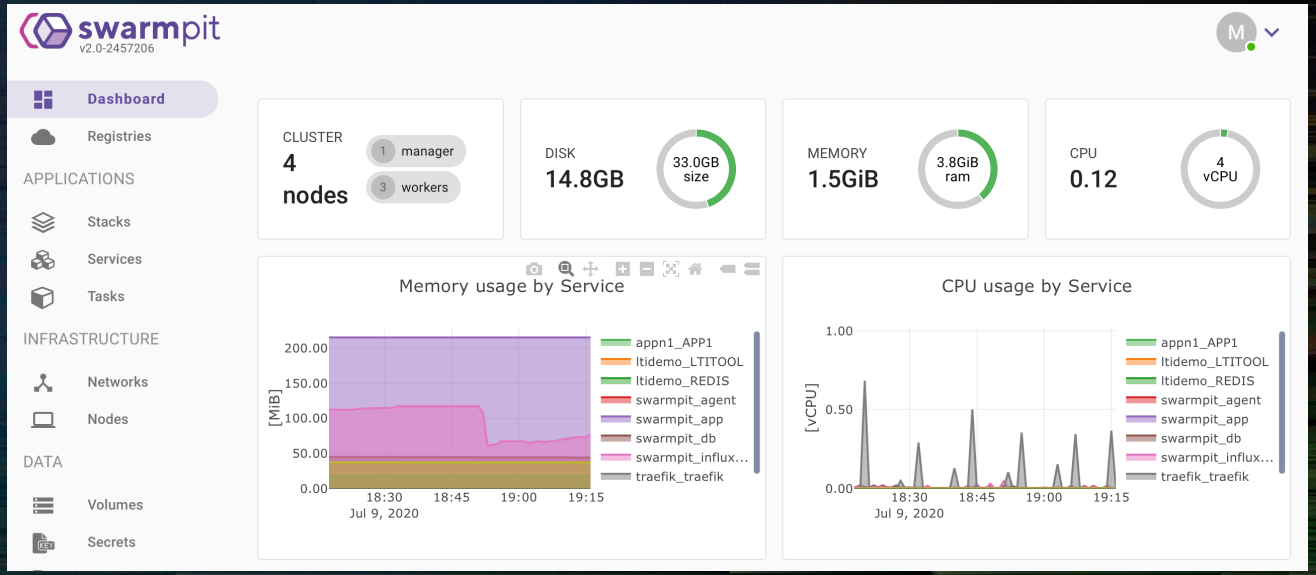


Service Inspection and Management

https:// swarmpit.io, https://github.com/swarmpit and https://dockerswarm.rocks/swarmpit/

UX for Image, Stack, and Service management

Others:
Portainer
Rancher
Dockstation



Resources

Developers

Office hours: https://community.blackboard.com/groups/home/78
Techies slack channel: https://tinyurl.com/JoinBlackboardTechiesSlack
Documentation: https://docs.blackboard.com
REST APIs: https://developer.blackboard.com/portal/displayApi

Containers and Orchestration

Docker & Docks: https://docker.com & https://docs.docker.com
Docker Swarm: https://docs.docker.com/engine/swarm/
Docker Desktop: https://docs.docker.com/get-docker/
Ngrok: https://ngrok.com
Traefik: https://traefik.io & https://dockerswarm.rocks/traefik/
Swarmpit: https://swarmpit.io & https://dockerswarm.rocks/swarmpit/
Orchestration Tools: https://www.g2.com/categories/container-orchestration
Play with Docker: https://labs.play-with-docker.com/
Tutorials: https://tinyurl.com/Udemy-docker-mastery and https://www.freecodecamp.org (search for docker)

Examples

https://github.com/moneil/dockerdemo-nginx-certbot
https://github.com/moneil/dockerdemo
https://github.com/moneil/appN

More examples and documentation coming in 2021-2022!


BbWORLD 21
Blackboard
#BbWorld21