Unifying Log Aggregation in a Containerized World

Getting your shit in order with Docker Logs

We'll cover...

- How we should be thinking about and treating containers
- How the underlying technology for Docker works
- How we can take advantage of Docker to standardize our stack
  - Different approaches for different needs
  - Beyond volumes and logging drivers
  - When to to use which approach
  - Pros and cons to the various approaches
  - Some really powerful stuff third-party tools can provide

Background on Me

Brad Whitfield

- Site Reliability Engineer at Capital One
- Previously Cloud Engineer at Autodesk
- Running Docker in production for over two years
- I’ve tried many of these strategies
- I’m a Docker Certified Associate, if you like certs

- Married
- Mountain bike
- Rock climb
- Play video games
- Can’t spell containerized

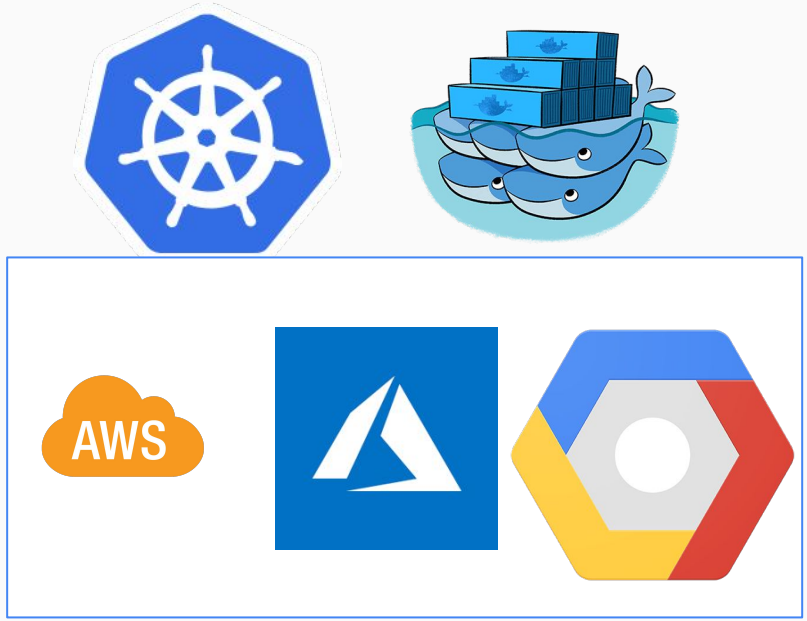
Containers - Processes, not VMs

Linux Containers (LXC)

- More like a package manager but with runtime isolation
- Images provide all the binary bits needed to run an application
- Containers are the runtime components of a process
  - Environment Variables
  - CPU Affinity
  - Memory
  - Networking
  - etc...

Orchestration

- Tools like Kubernetes and Swarm bring the control plane up a level above the cloud provider
- Sysadmins have a chance to provide uniformity/standards to apps by building monitoring, logging, and alerting into the platform
- Developers can ship a company's competitive advantage faster



Docker Deep Dive

Process ID (PID) 1

- Docker and LXC uses Linux namespace isolation
- PID namespaces prevent processes from seeing other processes on a host
- A container starts with its own PID 1
- Docker only tracks PID 1 and its children
- A containers entrypoint/cmd is typically PID 1

STDOUT and STDERR

- STDOUT is standard out and STDERR is standard error
- Docker acts like systemd in some ways
  - Journald captures all STDOUT and STDERR for processes managed by systemd
- Docker captures the STDOUT and STDERR of all processes it tracks
  - Applies to PID 1 and children of it
  - Think printing to a terminal
- The output is streamed to the configured logging driver (json by default)

Demo

https://github.com/bradwhitfield/logging-talk/tree/master/pid1-stdout

The Docker Socket

- Docker is a CLI and a Daemon
- CLI makes RESTful calls to the daemon
- Daemon interprets the calls and runs containers
- You can see evidence of this if you've used Docker Machine.

Demo

https://github.com/bradwhitfield/logging-talk/tree/master/docker-socket

Location on Disk

- When no logging driver is configured, Docker writes the output to json files
- Usually at `/var/lib/docker/containers/containerId/contianerId-json.log`
- `docker inspect` can show the location

```
> docker inspect httpd -f '{{ .LogPath }}'
/var/lib/docker/containers/b162d1dfd6c1e37f9b85054677b48d74be47ea8b430e1b9b7ecf310d0a7b24e0/b162d1dfd6c1e37f
9b85054677b48d74be47ea8b430e1b9b7ecf310d0a7b24e0-json.log
```

Logging Drivers

- Docker has built-in drivers that ship your logs to various platforms
- Default is JSON
- Only JSON and Journald allow the `docker logs` command to work
- We'll cover more in a moment

Container Level Log Aggregation

Logging Drivers - Easy Wins

- The simplest method to implement
- Docker has built-in drivers that ship your logs to various platforms
  - AWS CloudWatch Logs
  - GCP StackDriver Logs
  - GELF (Logstash)
  - Fluentd
  - etc…
- All STDOUT and STDERR are shipped to the configured logging driver
- Configure in your run commands
- https://docs.docker.com/config/containers/logging/configure/

Logging Drivers - Pros and Cons

Pros

- Built in to Docker
- Really easy to get started with
- Can be configured at the host level
- Great if you only have a few containers

Cons

- They don’t scale well (think 100s of containers)
- Have to be reconfigured on all containers if your logging platform changes
- If your logging infrastructure is down, containers won’t start

Demo

https://github.com/bradwhitfield/logging-talk/tree/master/logging-drviers

Volumes - If You Must...

- Map static files to disk
- Use a third-party tool to ship logs
- The inspiration for this talk

Volumes - Pros and Cons

Pros

- Can be used with non-cloud native applications that only log to file
- Maybe some other niche use cases

Cons

- Managing state with volumes is complex
- Portable volumes are complex
- Harder to manage host disk space
- Metadata from the container is not included
- Volume per container

In Code

- A logging library may handle writing to logging infrastructure
- The logging infrastructure is tightly coupled to the application code/configuration

Please Don’t Do This

Host Level Log Aggregation

Logging Drivers - Still an Easy Win

- The simplest method to implement
- All available logging drivers can be configured in Docker's `/etc/daemon.json`
- Some logging drivers have more options when set at the daemon level
- STDOUT and STDERR of all containers are shipped to the configured logging driver
- Can be overridden at the container level (run command)
- https://docs.docker.com/config/containers/logging/configure/

Logging Drivers - Pros and Cons

Pros

- Built in to Docker
- Really easy to get started with
- Can be easily overridden
- Configured once

Cons

- If your logging infrastructure is down, containers won't start

Third-Party Utilities - Really Powerful Stuff

- Install on the host or as containers
- Configure as a DaemonSet in Kubernetes
- Configure as a Global service Swarm
- Examples:
  - Fluentd
  - Beats
  - Splunk
  - Logspout

splunk>

Third-Party Tools - Pros and Cons

Pros

- Metadata is added to log events*
- Can forward to multiple locations*
  - Make migrating logging infrastructure easier (Splunk to Elastic is common)
- Great for internal PaaS
- Logs can be accessed from hosts
- Configured once
- Can help enforce standards

Cons

- More disk space is required
- Initial setup is more complex
- Infrastructure is more complex
- Mounting the Docker socket is a security concern to consider

* Depends on the tool

Demo

https://github.com/bradwhitfield/logging-talk/tree/master/docker-elk

Third Party Utilities - Notes

- Barely scratching the surface of what's possible
- Customize your process to fit your business needs
- ProTip: Configure your default logging driver to cap disk space
- Many tools can aggregate Kubernetes specific metadata
- https://www.splunk.com/blog/2015/08/24/collecting-docker-logs-and-stats-with-splunk.html
- https://github.com/fabric8io/fluent-plugin-kubernetes
- https://github.com/gliderlabs/logspout
- https://www.elastic.co/blog/enrich-docker-logs-with-filebeat

More info and slides

https://bradwhitfield.com/posts/log-aggregation-with-docker
(soon)
https://bradwhitfield.com/log-slides.pdf

Questions?