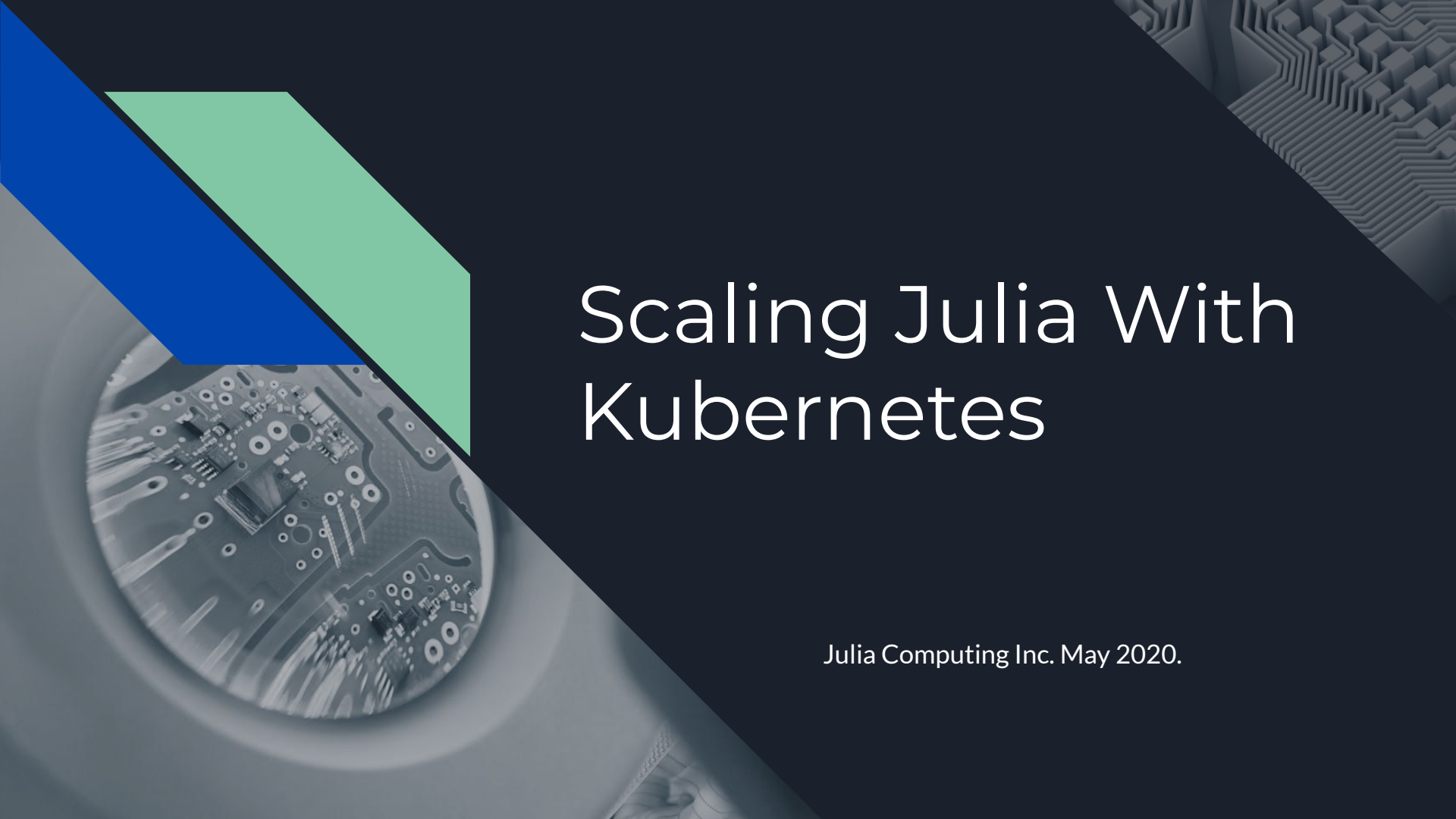


Scaling Julia With Kubernetes

Julia Computing Inc. May 2020.

TOC

What is Kubernetes

- Container orchestration tool
- Automate container deployment, scaling and management
- Open and unified API
- Public, Private and Hybrid
  - EKS, GKE, AKS
  - OpenShift
- https://kubernetes.io/

Why Kubernetes

- Benefits of containerization
  - Security: Isolation
  - Efficiency: Isolation at a lower cost compared to VMs
  - Agility: Seamless dev-test workflow
  - Portability: Packaged environment for your application
- Ease of management, deployment, scaling
  - Depth and breadth of API coverage
- Flexibility: Run anywhere, same interface
  - Any cloud provider managed cluster
  - Cloud VMs - create your own cluster
  - In -house cluster

Using Kuber.jl

- https://github.com/JuliaComputing/Kuber.jl

- Starting clusters
  - Managed cloud clusters are quite simple.
- Connecting to a cluster
  - Authentication and API versions
  - Examining the cluster itself
-
- REST API paradigm - entities and verbs
  - get/list, put, update!, and delete!
-
- The Container Image Repository
  - Fetch images
  - Build images

Notebook 1: Using Kuber.jl Package

Launching Simple Applications

- Run an application
  - Run nginx
  - Looking up service endpoints
  - Accessing services
  - Getting into the container
    - Look at logs
    - Show ngxtop

- Deploy Julia application
- Assigning resources

Notebook 2: Launching Simple Applications

Notebook 3: Running Julia in Kubernetes

Launching Parallel Julia Tasks

- Types of program execution
  - Threads
  - Processes
- Choosing between Threads and Processes

- Scaling
  - Larger (single) machine (Vertical)
  - Multiple machines (Horizontal)

Parallel Julia - Vertical Scaling

Single Pod/Container

- Run Julia Process with multiple threads

- Run Multiple Julia processes
  - Connected (Master - Worker)
  - Independent (Just workers)

- Multiple processes, each with multiple threads

Notebook 4: Running Parallel Julia - Vertical Scaling

Parallel Julia - Horizontal Scaling

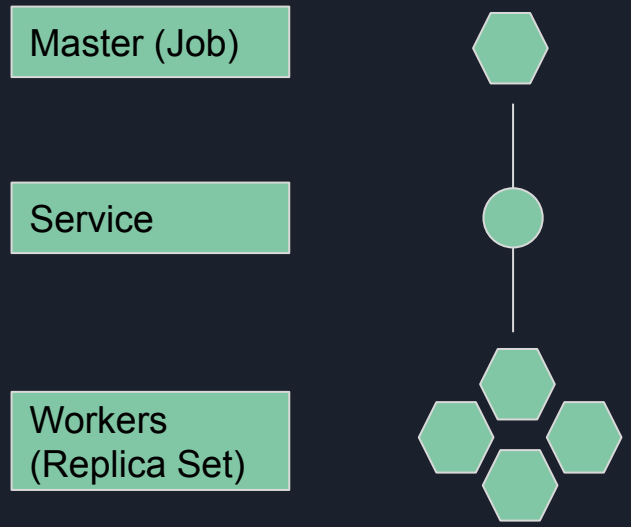
Multiple containers, likely across machines

- Independent Workers
- Master-Worker across machines
  - Needs networking to be setup

- Can not rely on having access to the same file system
- Processes get their own network host name

Parallel Julia - Mapping to Kubernetes

- Relevant Kubernetes Entities
  - Job
  - Replica Set
  - Service



Notebook 5: Running Parallel Julia - Horizontal Scaling

More Patterns & Techniques

- Queuing & Partitioning
  - Distributed stdlib - pmap, @distributed, Channels
  - Others: RabbitMQ, Redis
- Auto scaling by monitoring metrics
  - inbuilt metrics for cpu and memory
  - Third party tools (e.g. Prometheus) for App specific metrics
- Updating applications on the go (rolling updates)

JuliaTeam and JuliaRun

- Enterprise governance
- Private package management
- IDE integration
- Deployment and scalability
- Recordings of past Webinars on JuliaTeam and JuliaRun: https://juliacomputing.com/resources/webinars

Thank you!

Q & A