KubeCon | CloudNativeCon

North America 2021

RESILIENCE REALIZED

Container Checkpoint/Restore at Scale for Fast Pod Startup Time

Ritesh Naik

About Me

Ritesh Naik, MathWorks

- Senior Software Engineer
- Passionate about distributed system and cloud native applications

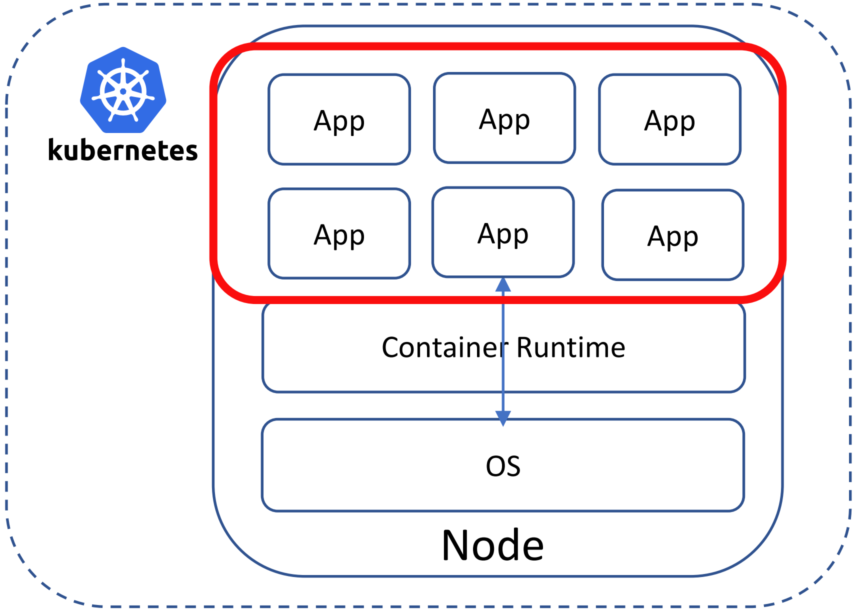
riteshnaik

✉ **rnaik@mathworks.com**

During this talk…

- Motivation
- Checkpoint/Restore introduction
- Checkpoint/Restore in Kubernetes Demo
- Q/A

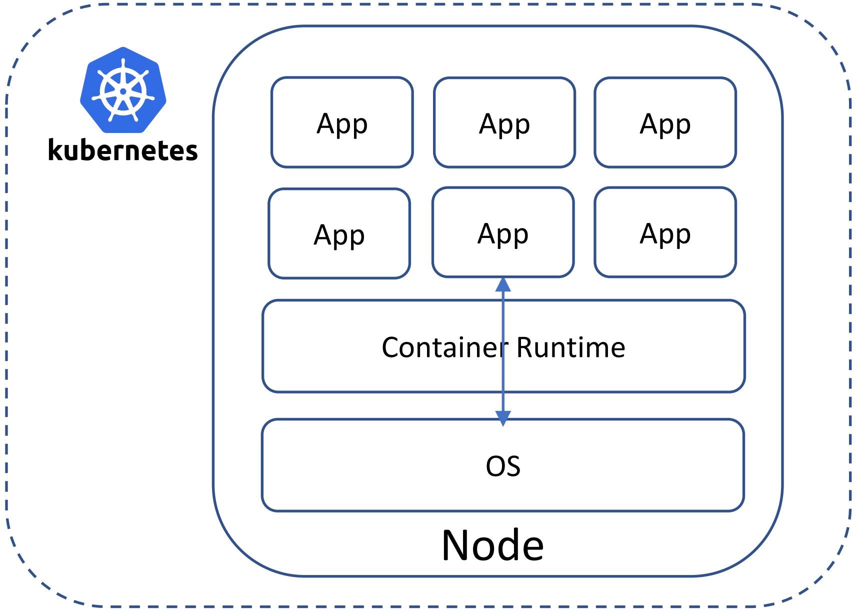
Our goal is to create a scalable system…



Our Goals:

1. Fast scale out time (for bursty workloads)
2. Fast performance on first use of a pod or container (no cold start pains)
3. Great utilization (low-cost waste)

…But achieving all three goals is challenging



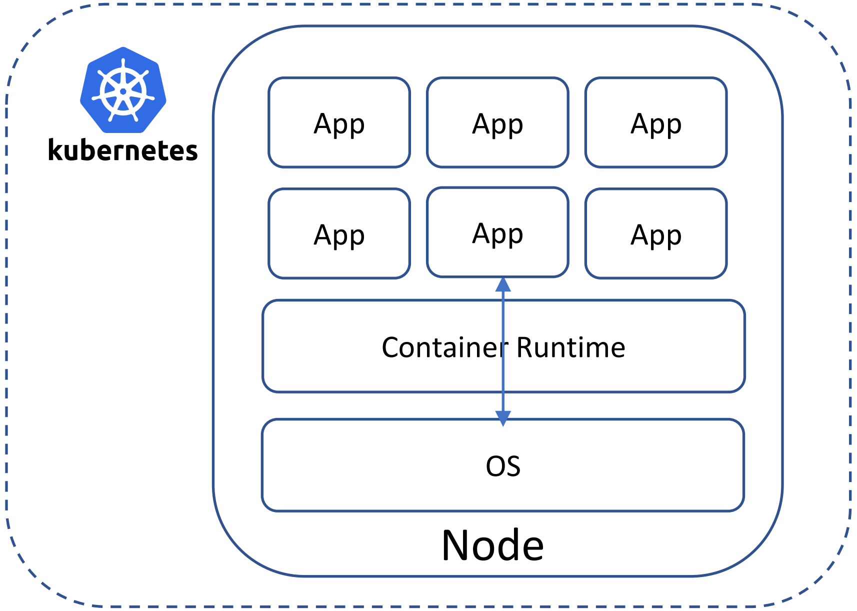
Our Goals:

1. Fast scale out time (for bursty usage)
2. Fast performance on first use of a pod or container (no cold start pains)
3. Great utilization (low-Cost waste)

The Challenge:

- Container/application cold start time makes it difficult to get all three goals at the same time
- Note: A similar challenge may also come up in FaaS/Serverless use cases

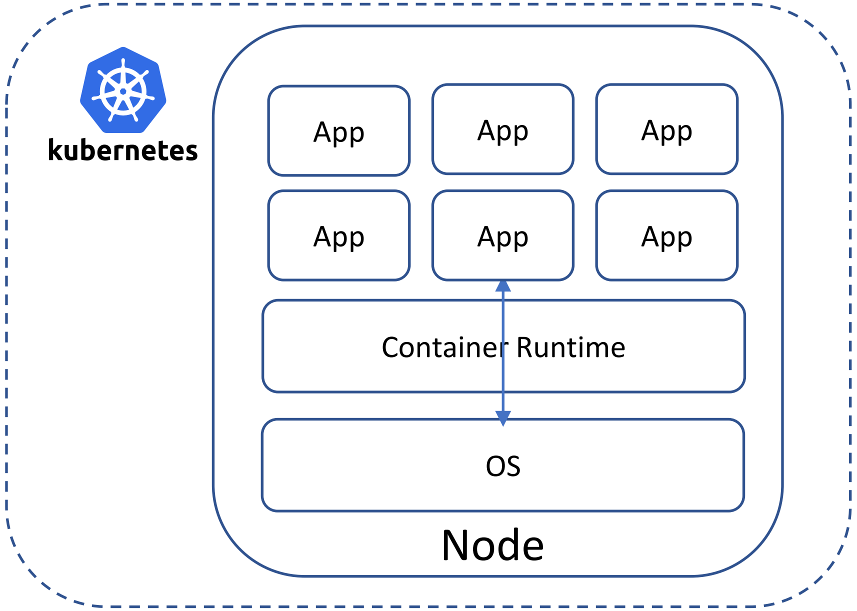
Option 1: Scale out on-demand with usage



	On-Demand Scaling
Fast performance on first use of a pod or container (no cold start pains)	✗
Fast Scale Out Time (for bursty workload)	✗
Great utilization (low-cost waste)	✓

- Wait for the usage demand
- Scale up the workload

Option 2: Create pre-warmed standby pool



- Maintain a pool of pre-warmed containers
- Size of the pool could be calculated based on the historical traffic trend

	Pre-Warm Standby Pool
Fast performance on first use of a pod or container (no cold start pains)	✓
Fast Scale Out Time (for bursty workload)	✗
Great utilization (low-cost waste)	✗

Is there a way to achieve all goals?

	On-Demand Scaling	Pre-Warm Standby Pool	???
Fast performance on first use of a pod or container (no cold start pains)	✗	✓	✓
Fast Scale Out Time (for bursty workload)	✗	✗	✓
Great utilization (low-cost waste)	✓	✗	✓

Yes, we can!!! Checkpoint/Restore

	On-Demand Scaling	Pre-Warm Standby Pool	Checkpoint/Restore
Fast performance on first use of a pod or container (no cold start pains)	✗	✓	✓
Fast Scale Out Time (for bursty usage)	✗	✗	✓
Great Utilization (Low-Cost Waste)	✓	✗	✓

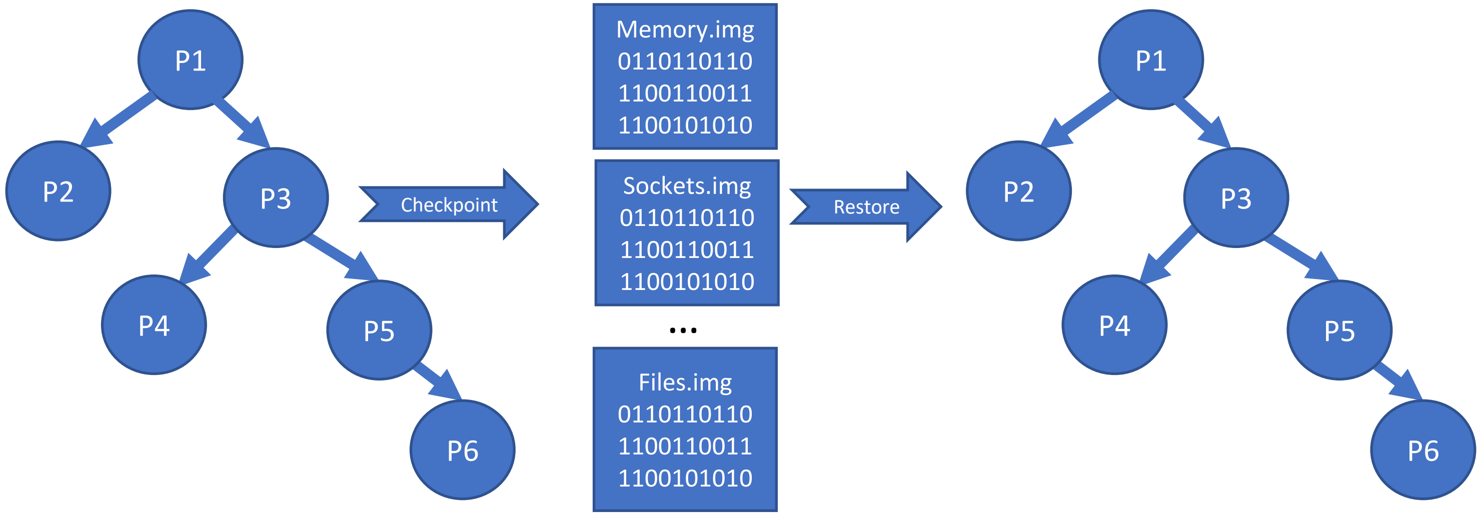
Checkpoint/Restore:

- Leverage CRIU project (Checkpoint Restore in Userspace)
- We can achieve all three goals by trading off a little on complexity
- Take runtime snapshot of process state of warmed container
- Restore as needed for scale

Checkpoint/Restore: Behind the Scenes

Checkpoint:

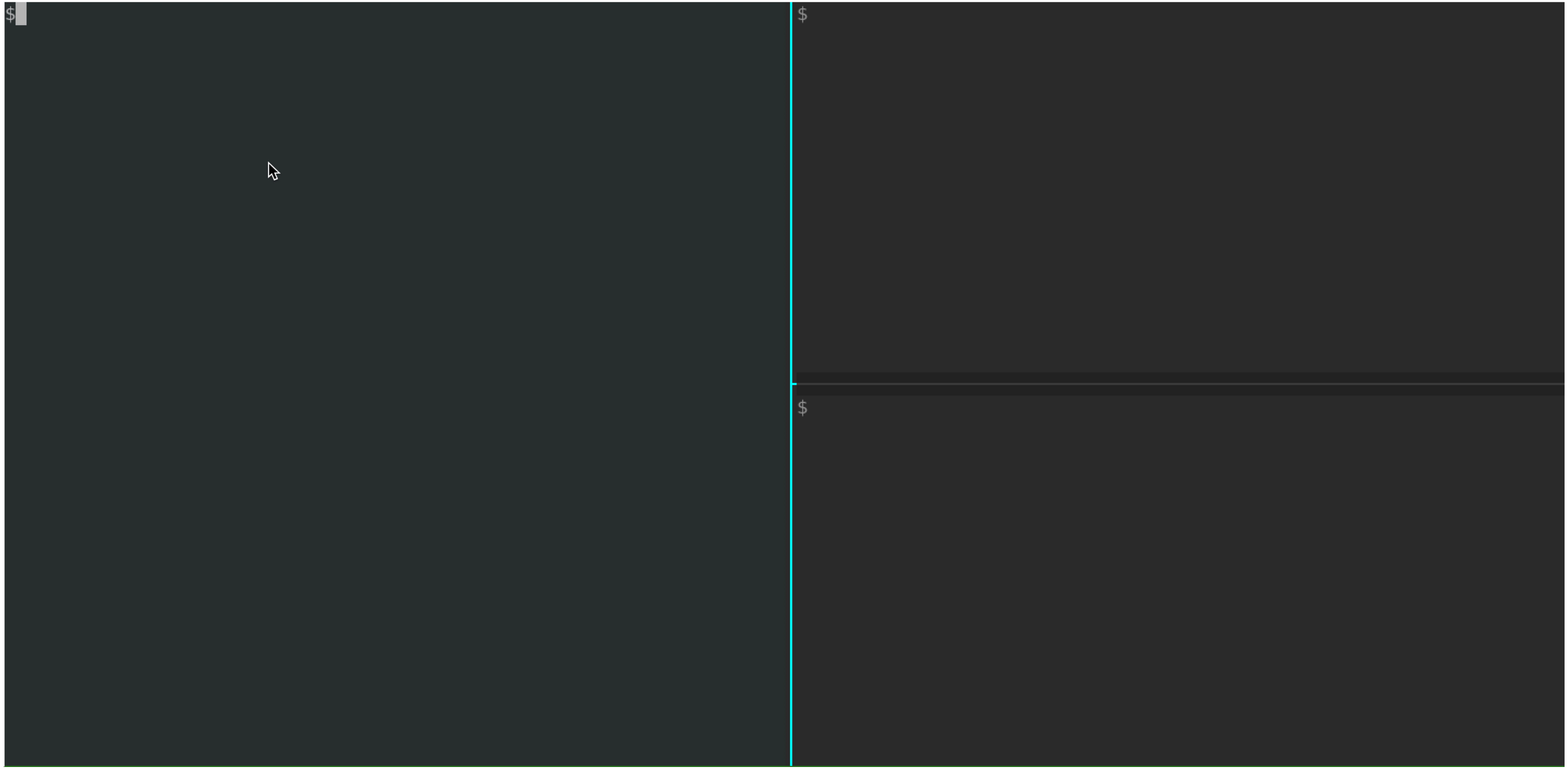
- Freeze process tree (i.e. ptrace)
- Collect memory contents, sockets, & other state (i.e. read /proc)
- Serialize state to image files



Restore:

- Read image files
- Create process tree scaffold (i.e. fork)
- Restore basic process state (fds, sockets, namespaces, cwd)
- Restore other process state (memory, timers, credentials)

Demo: Checkpoint/Restore of Containers

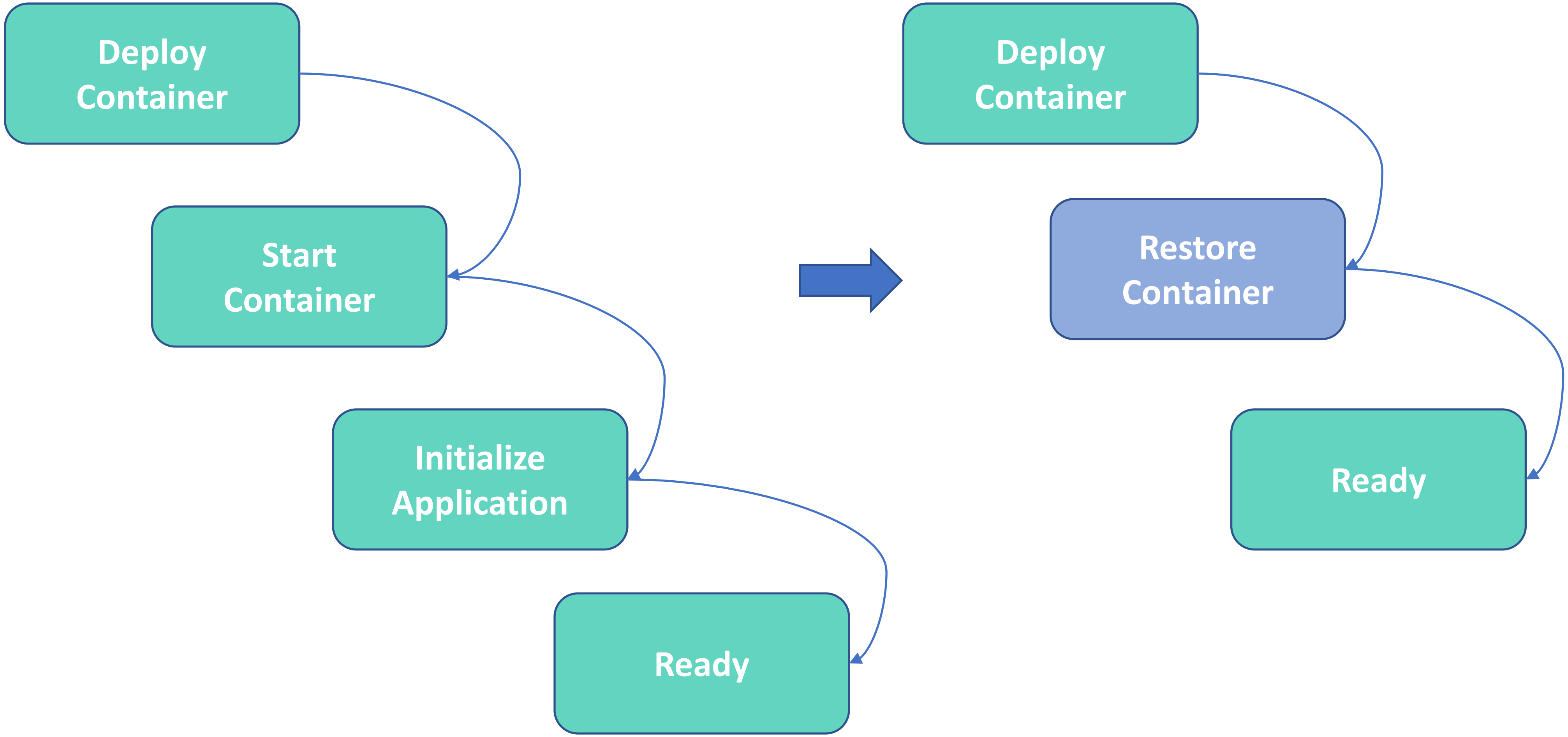


```
$
```

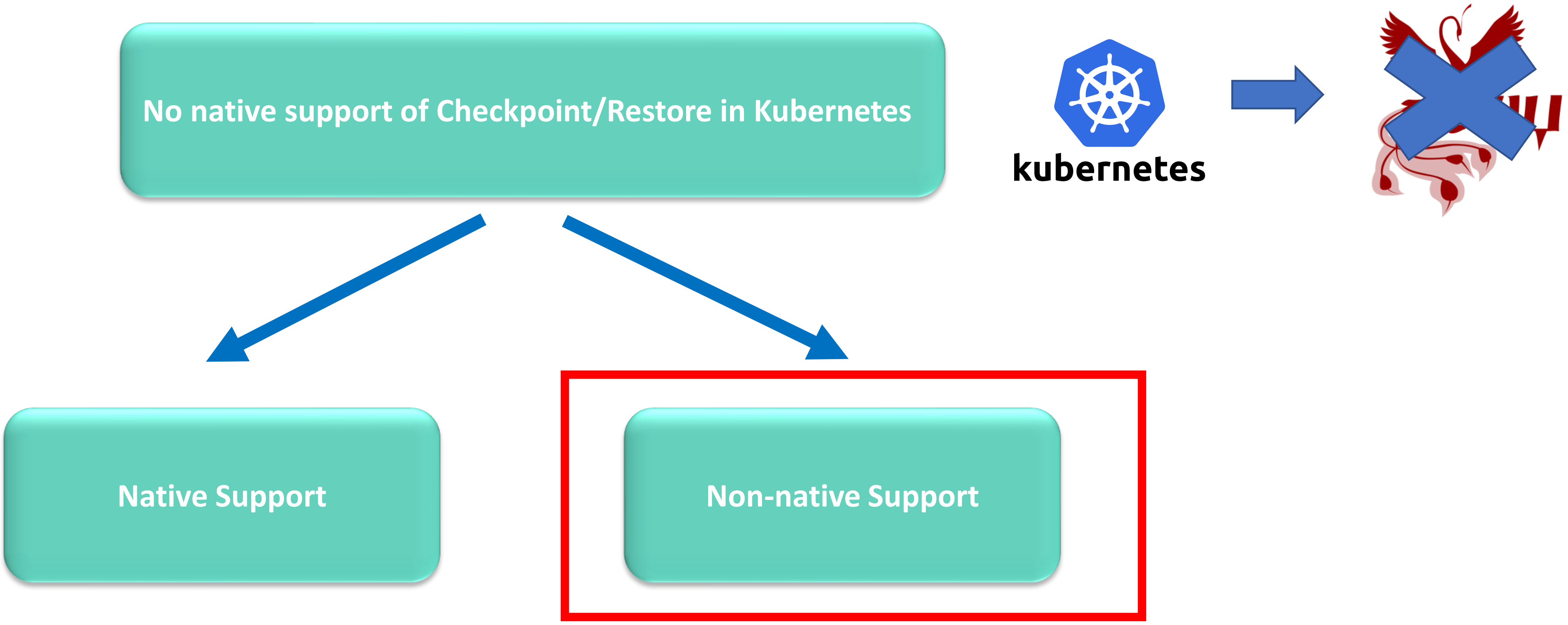
```
$
```

```
$
```

Eliminating Cold Starts with Checkpoint Restore



Kubernetes and Checkpoint/Restore

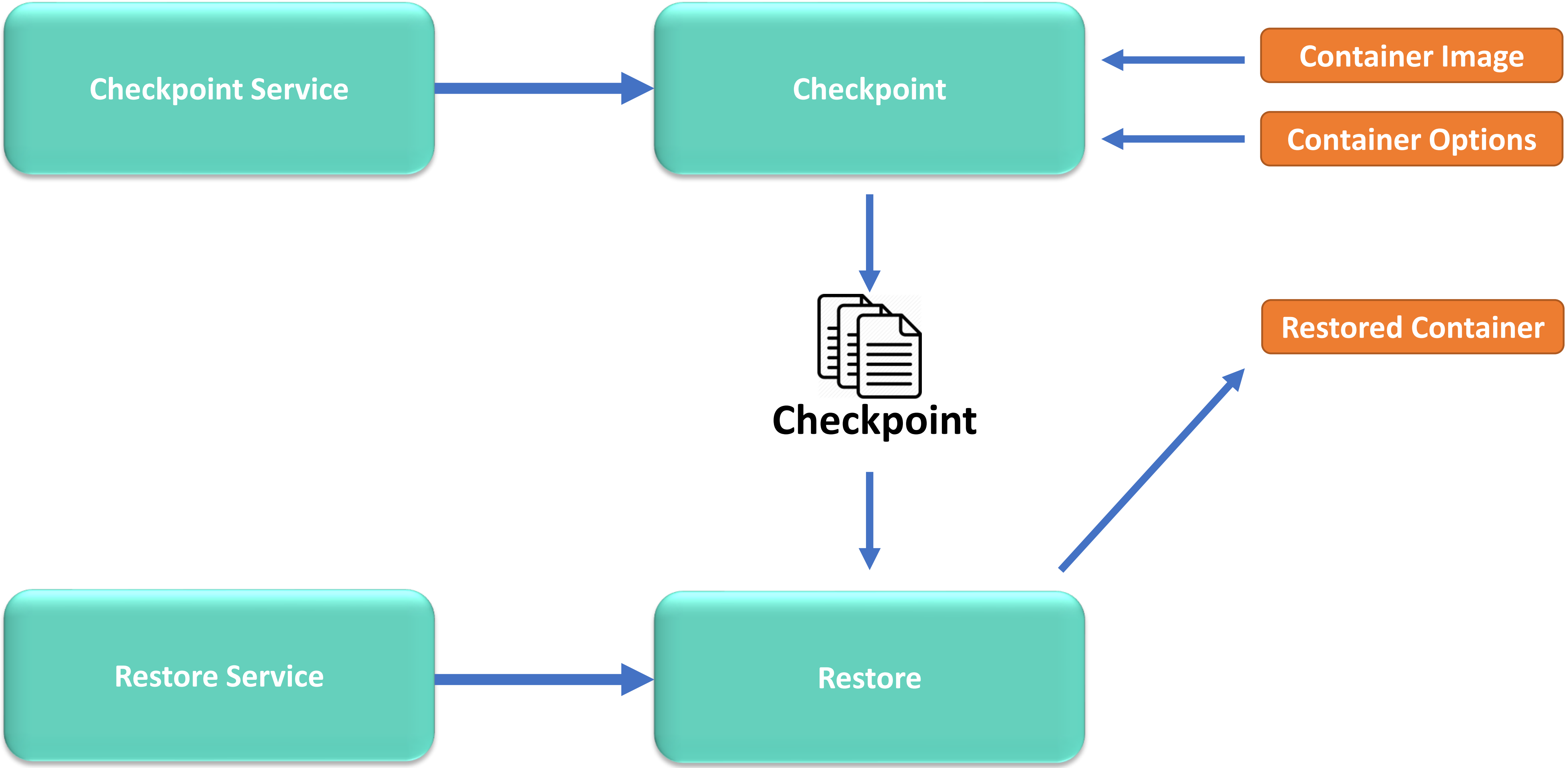


No native support of Checkpoint/Restore in Kubernetes

Native Support

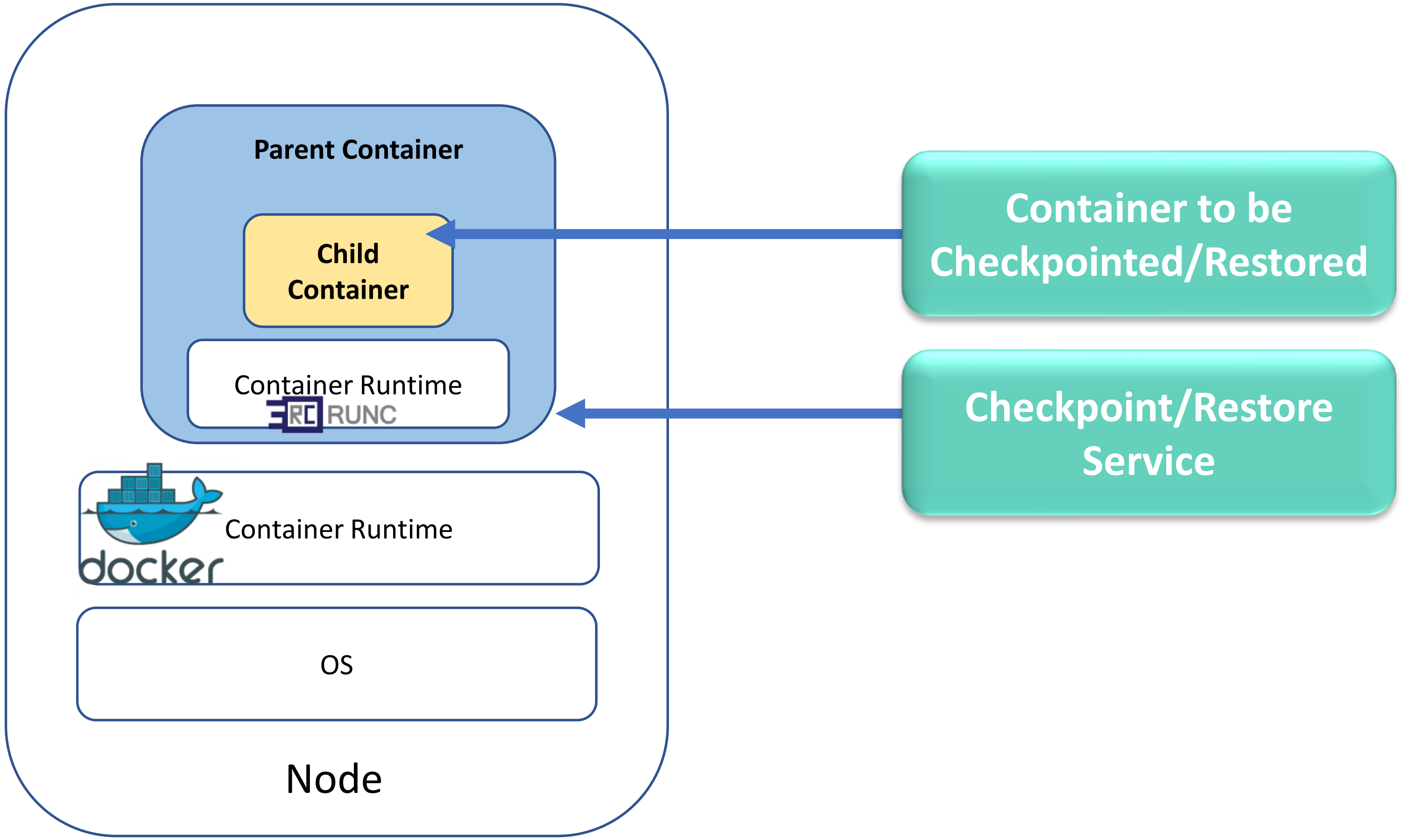
Non-native Support

Our Approach of Non-Native support

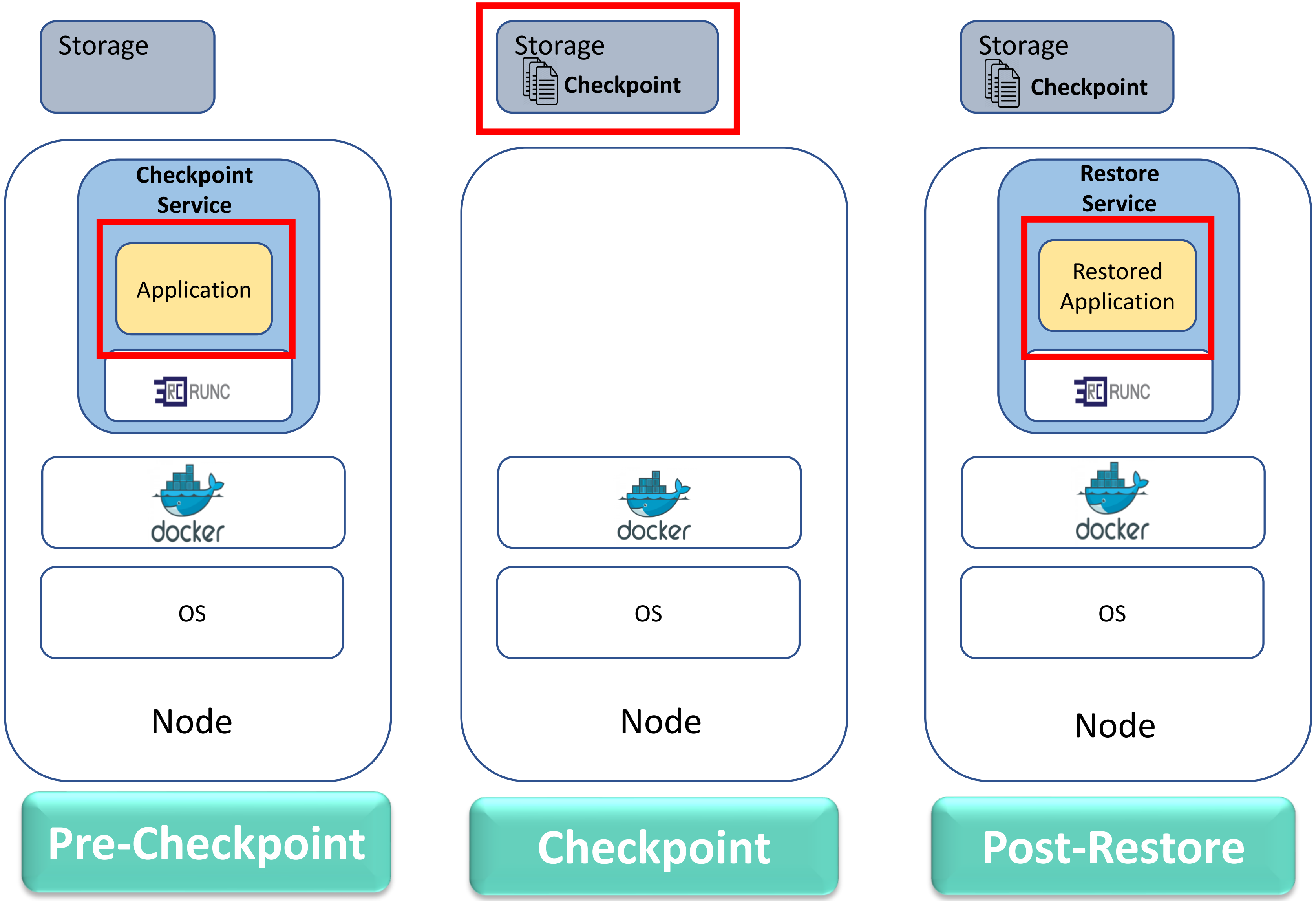


Container Runtime in Container Runtime

runc-in-docker



Container Runtime in Container Runtime



Storage

Checkpoint Service

Application

RUNC

docker

OS

Node

Pre-Checkpoint

Storage

Checkpoint

docker

OS

Node

Checkpoint

Storage

Checkpoint

Restore Service

Restored Application

RUNC

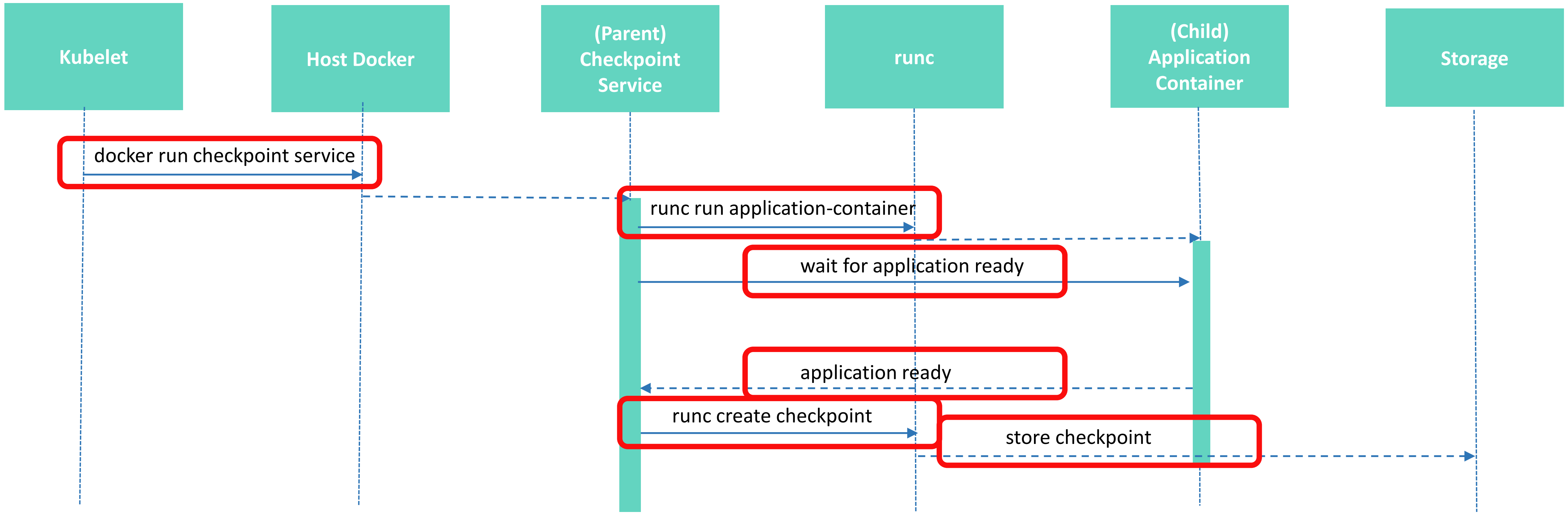
docker

OS

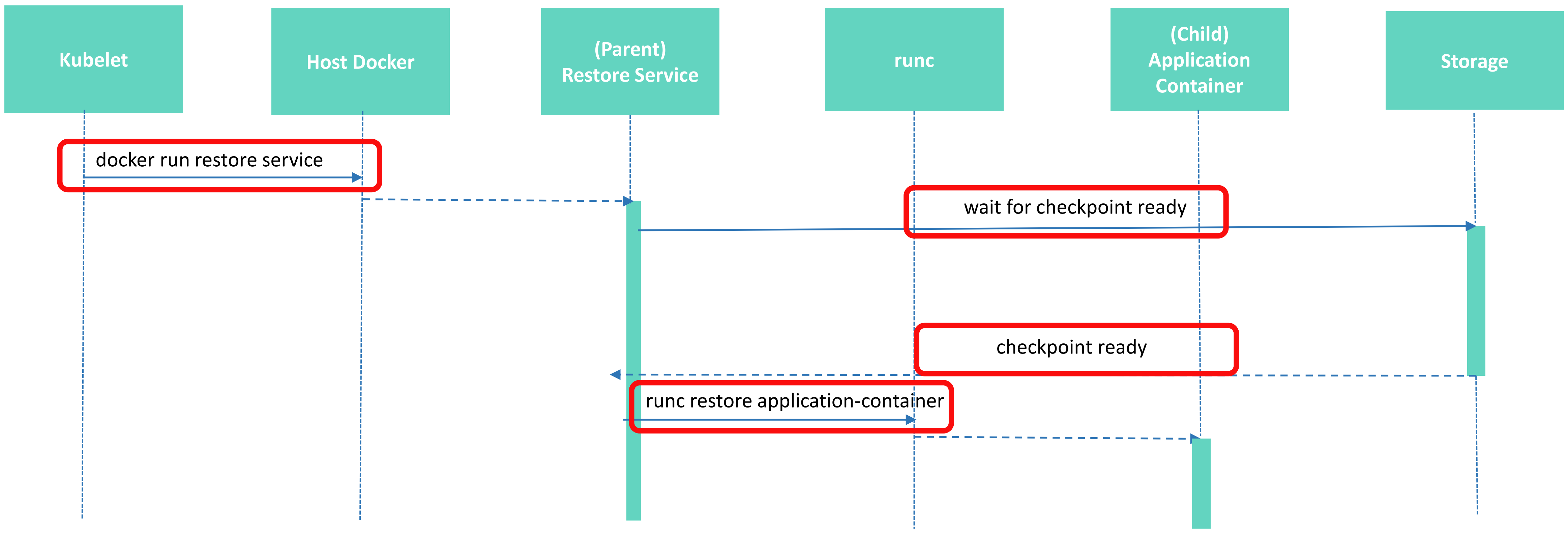
Node

Post-Restore

Checkpoint sequence

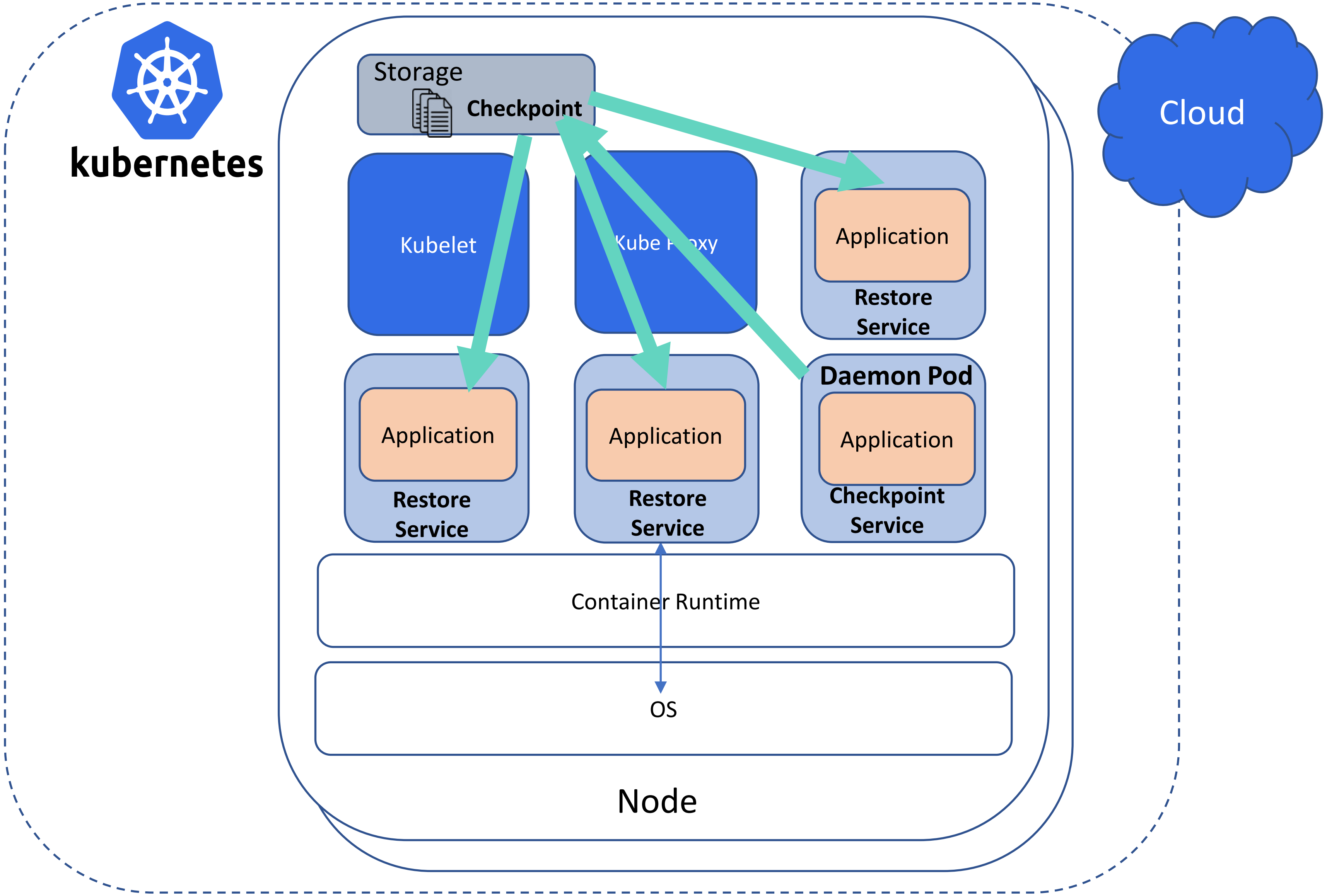

Kubelet
Host Docker
(Parent) Checkpoint Service
runc
(Child) Application Container
Storage
docker run checkpoint service
runc run application-container
wait for application ready
application ready
runc create checkpoint
store checkpoint

Restore Sequence

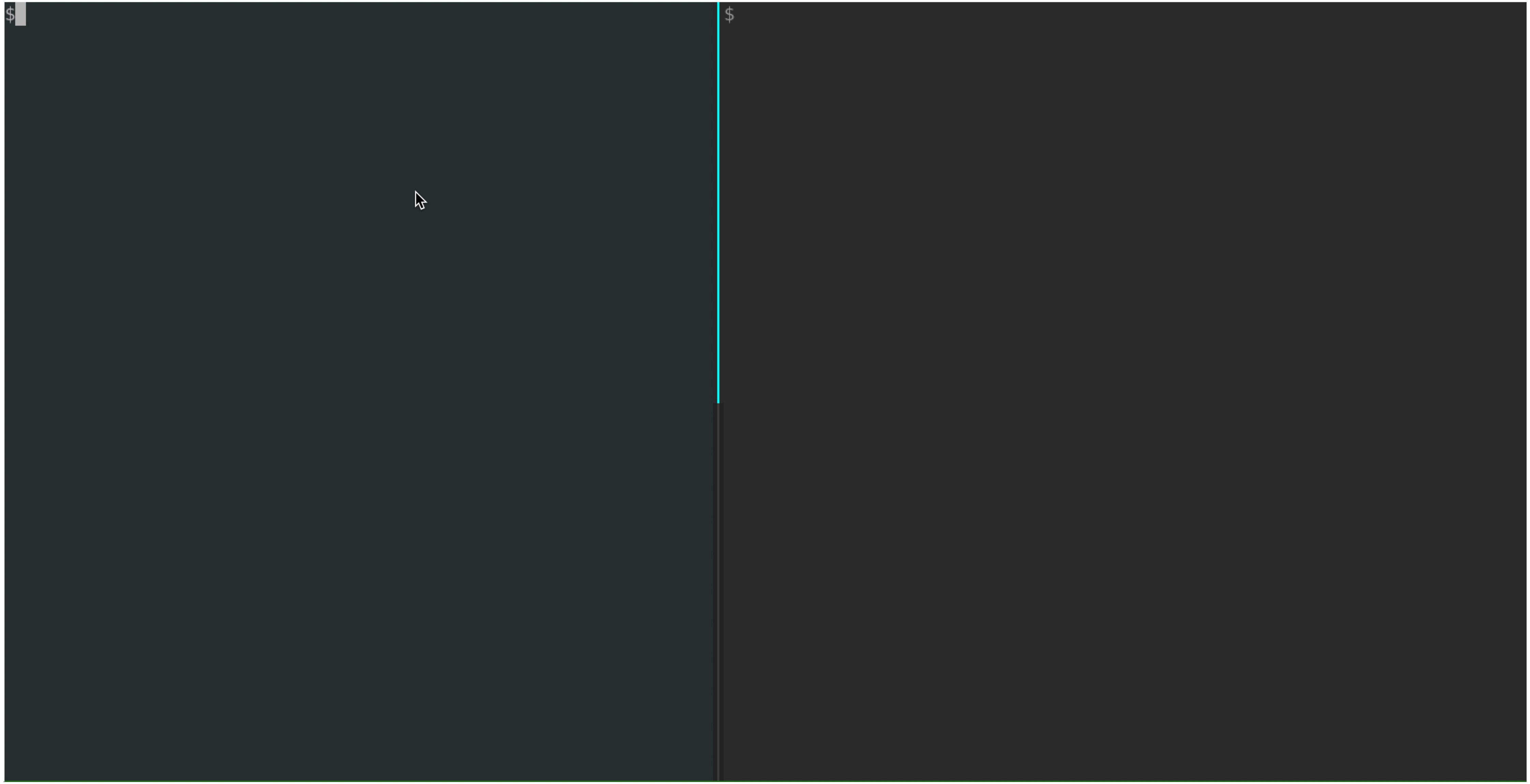


Kubelet
Host Docker
(Parent) Restore Service
runc
(Child) Application Container
Storage
docker run restore service
wait for checkpoint ready
checkpoint ready
runc restore application-container

Cluster Architecture with CR



Checkpoint Restore Service: Demo



```
$
```

```
$
```

Zero to Checkpoint Restore in Kubernetes

```
FROM ubuntu:18.04
CMD ["/app/main"]
```

```
FROM runc-base-image:latest as build
FROM go_app:latest
COPY --from=build /runc /runc
COPY config.json.template /runc/container/
USER root
CMD ["/bin/sh", "-c", "/runc/run.sh"]
```

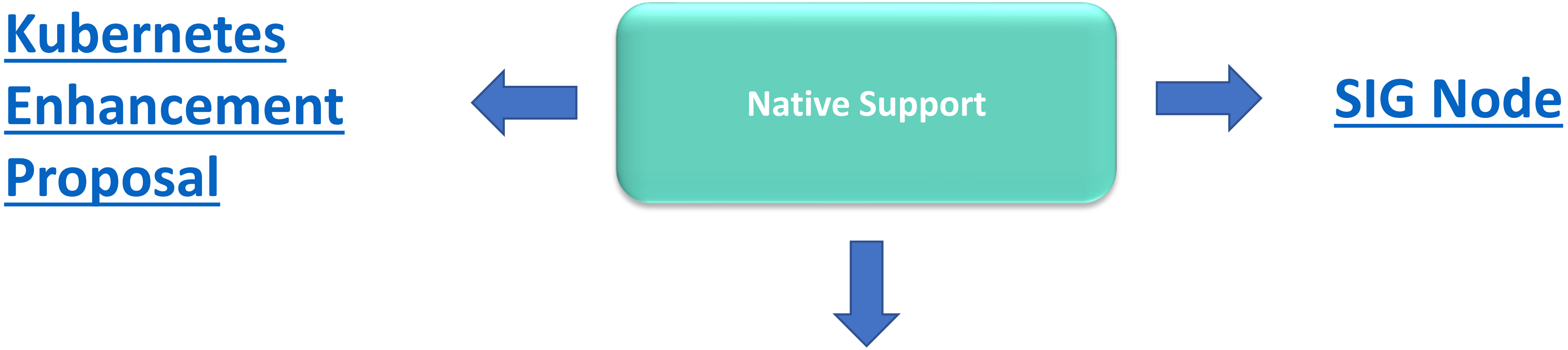
```
…
kind: Deployment
…
  spec:
    containers:
    - name: application
      image: go_app:latest
…
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: cr-demo
data:
  cr-properties: |
    cr.checkpoint.name=checkpoint
    cr.checkpoint.directory=demo
    cr.container.readiness=curl -s -o /dev/null localhost:4000
```

```
…
kind: Daemonset
…
  spec:
    containers:
    - name: checkpoint
      image: go_app_checkpoint:latest
      securityContext:
        privileged: true
      volumeMounts:
        - name: checkpoint-restore
          mountPath: /cr
        - name: cr-properties
          mountPath: /etc/cr.properties
          subPath: cr.properties
     volumes:
        - name: checkpoint-restore
          hostPath:
              path: /tmp
              type: Directory
        - name: cr-properties
          configMap:
              name: cr-demo
              items:
                 - key: cr-properties
                   path: cr.properties
```

```
…
kind: Deployment
…
  spec:
    containers:
    - name: restore
      image: go_app_restore:latest
      securityContext:
        privileged: true
      volumeMounts:
        - name: checkpoint-restore
          mountPath: /cr
        - name: cr-properties
          mountPath: /etc/cr.properties
          subPath: cr.properties
     volumes:
        - name: checkpoint-restore
          hostPath:
              path: /tmp
              type: Directory
        - name: cr-properties
          configMap:
              name: cr-demo
              items:
                 - key: cr-properties
                   path: cr.properties
```

Kubernetes and Checkpoint/Restore

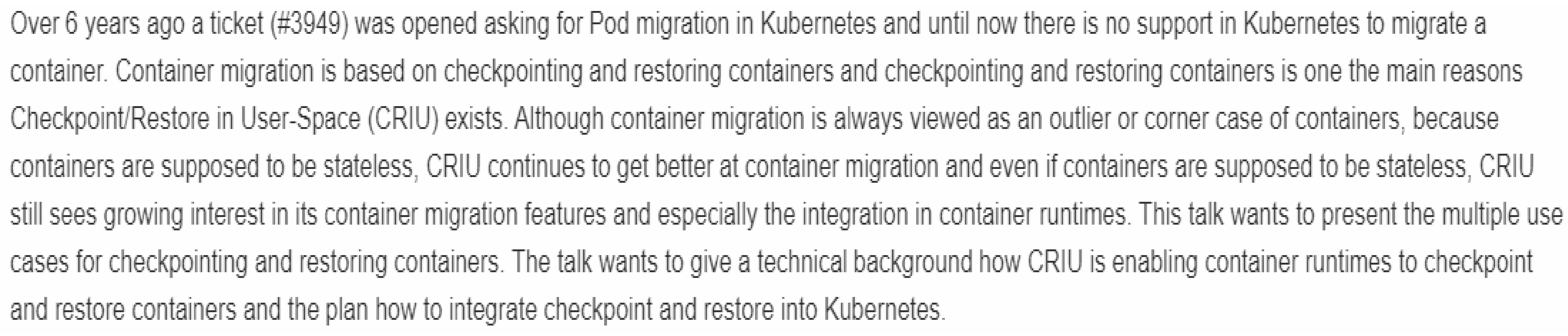


Kubernetes Enhancement Proposal

Native Support

SIG Node

Kubernetes and Checkpoint/Restore – Adrian Reber, Red Hat
Thursday, October 14 • 11:00am - 11:35am



Over 6 years ago a ticket (#3949) was opened asking for Pod migration in Kubernetes and until now there is no support in Kubernetes to migrate a container. Container migration is based on checkpointing and restoring containers and checkpointing and restoring containers is one the main reasons Checkpoint/Restore in User-Space (CRIU) exists. Although container migration is always viewed as an outlier or corner case of containers, because containers are supposed to be stateless, CRIU continues to get better at container migration and even if containers are supposed to be stateless, CRIU still sees growing interest in its container migration features and especially the integration in container runtimes. This talk wants to present the multiple use cases for checkpointing and restoring containers. The talk wants to give a technical background how CRIU is enabling container runtimes to checkpoint and restore containers and the plan how to integrate checkpoint and restore into Kubernetes.

Lessons Learned

- Checkpoint/Restore behavior is sensitive to changes in the process tree

- Checkpoint/Restore failure could be due to issues in different layers of technological stack

- Avoid optimizing to the container runtime

- Trading off portability vs. optimizations

Best Practices

- Enhance observability by enabling logs and metrics around Checkpoint/Restore

- Shift left in the CI/CD pipeline for quicker detection of any failures

- Make sure to keep CRIU packages and auxiliary components up to date

- Know the limitations and boundaries of CRIU

Future Enhancements

- Make the checkpoint accessible centrally across nodes
- Make the checkpoint part of CI pipeline
- Extend the support to other use cases like Pod Migration
- Extend the native support in Kubernetes to include the fast pod startup time use case

MathWorks is hiring….

RESILIENCE REALIZED

KubeCon | CloudNativeCon

North America 2021