KubeCon | CloudNativeCon

North America 2018

kasten

Day 2 With Stateful Applications

Implementing a Data Protection Strategy

Vaibhav Kamra
@vaibhavkamra

Deepika Dixit
@deepikadixit

about us

Vaibhav Kamra

CTO & Co-Founder @ Kasten

https://github.com/kanisterio

Previously @ Dell EMC, Maginatics, Microsoft

@vaibhavkamra

Deepika Dixit

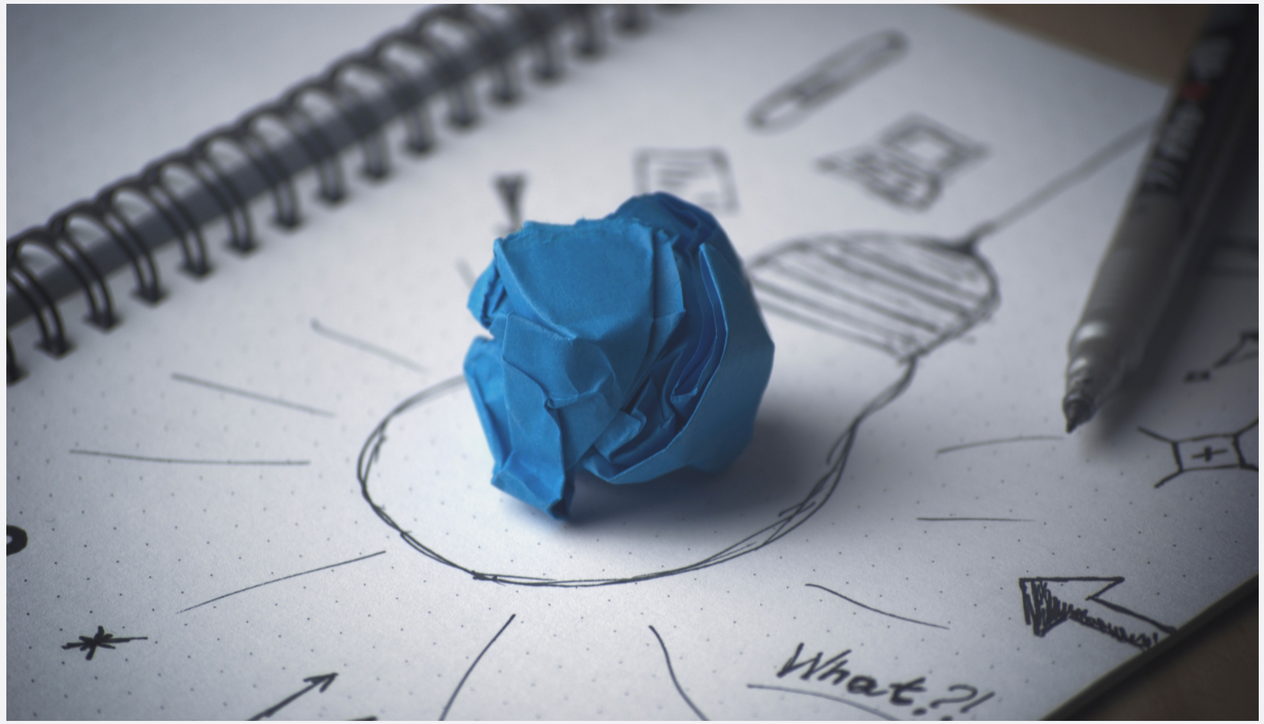
MTS @ Kasten

https://github.com/kanisterio

Previously @ Tintri, ASU

@deepikadixit

agenda
what we'll cover



Where is the Data?

Adoption patterns of Stateful Applications in Kubernetes

Data Protection Strategy

What, Why, Misconceptions

Getting it Right

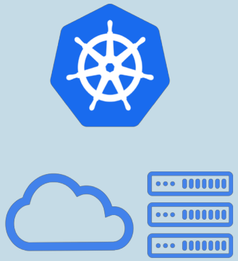
Implementing Data Protection in Kubernetes

Tools available

Demo

show of hands
where is the data

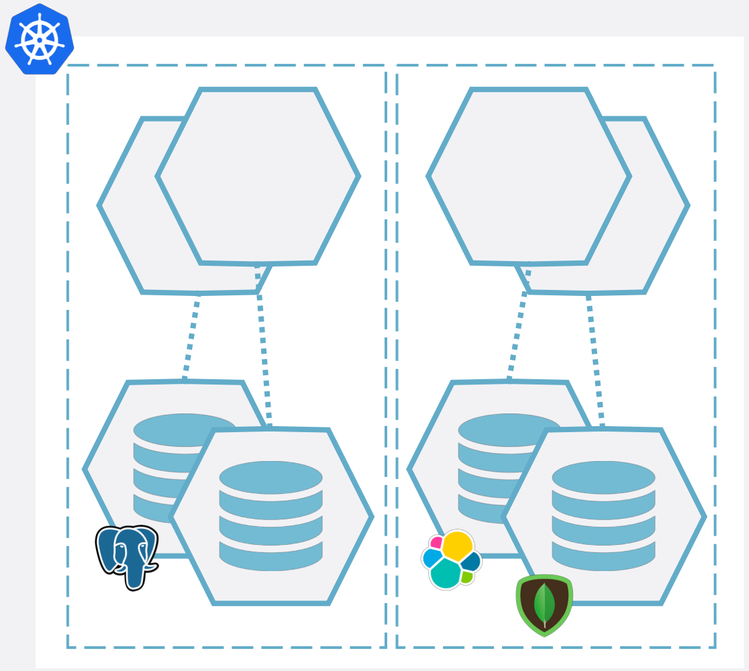
Who is running stateful applications in Kubernetes?



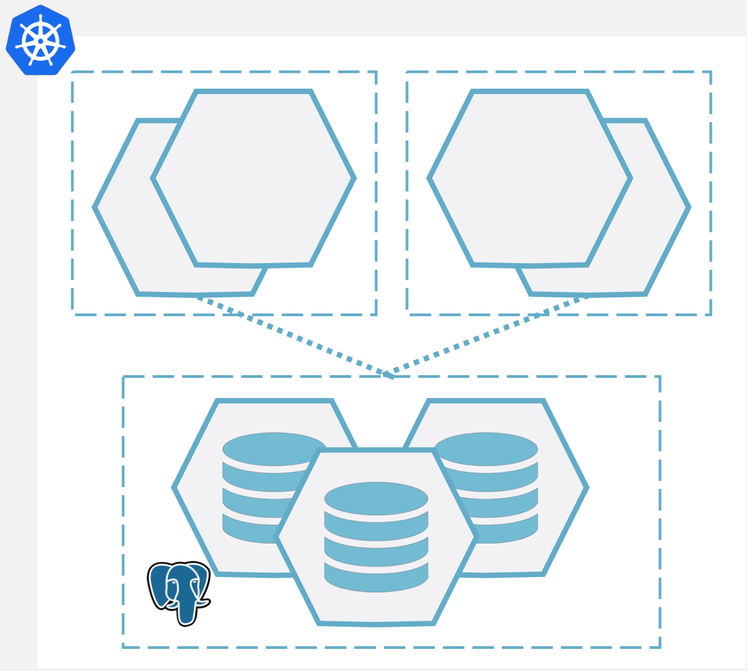
Who is running applications that store data in services outside of Kubernetes?

kubernetes stateful applications
wide variety of patterns

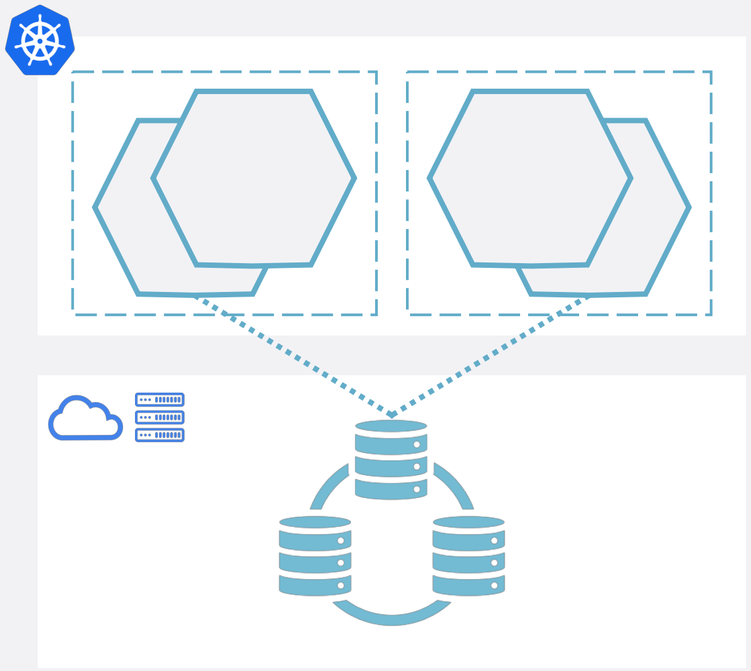
Application includes
data services – all in Kubernetes



Data services in Kubernetes –
separate from Application



Application uses
data services outside of
Kubernetes



data protection strategy
what and why

Systems in place to recover applications and data if things go bad

Accidental or Malicious Data Loss

Infrastructure or Hardware Failure

Application Misconfiguration

Regulatory Compliance

data protection strategy
key elements

Automated Backup and Recovery

Scheduling and Retirement Policies

Security and Encryption

Recovery SLAs

data protection strategy
key elements

Automated Backup and Recovery

Scheduling and Retirement Policies

Security and Encryption

Recovery SLAs

“Operate At Scale”

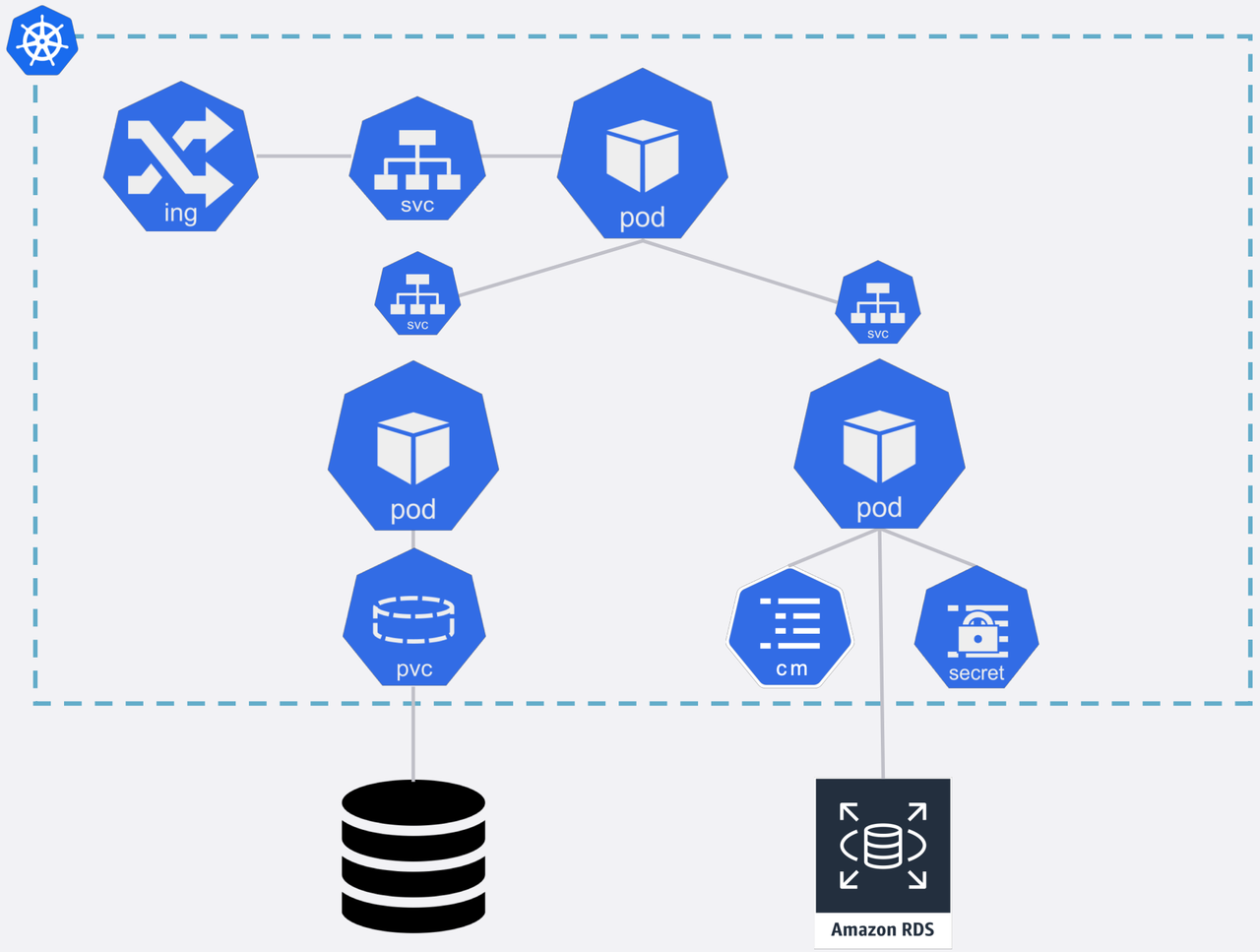
data protection strategy
misconceptions

"I don't have any Stateful Applications in Kubernetes"

"My data stores are replicated and resilient"

"My underlying infrastructure already takes care of this"

anatomy of a cloud-native app
kubernetes resources and persistent state

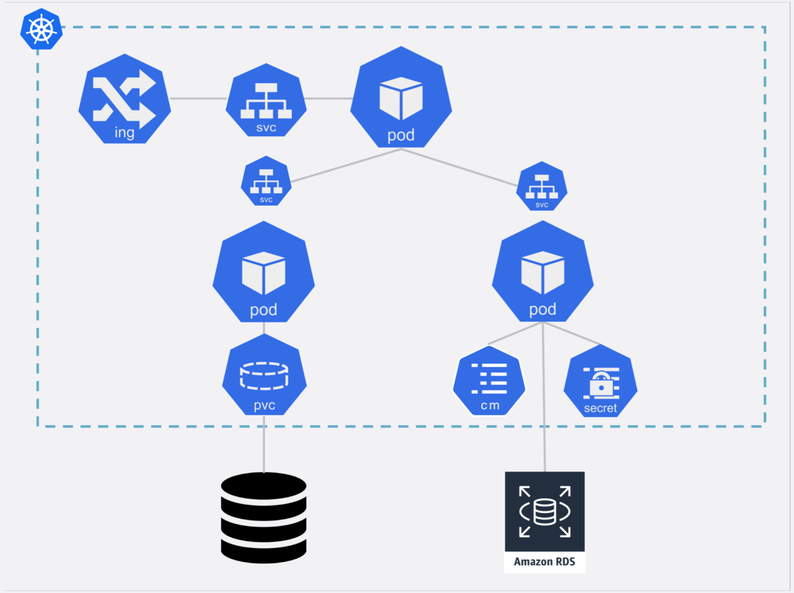


implementing data protection

implementation
capture application config

Application Definition (Kubernetes Resources)

- From Kubernetes API Server
- From Source Code (infra-as-code)
- From Helm Repo



Other State

- Pipeline state/Release information
- Environment config

implementation
capture persistent data

Unstructured Data from PVCs

- Volume Snapshots
- File System backups
- A combination of both

Data services in the application

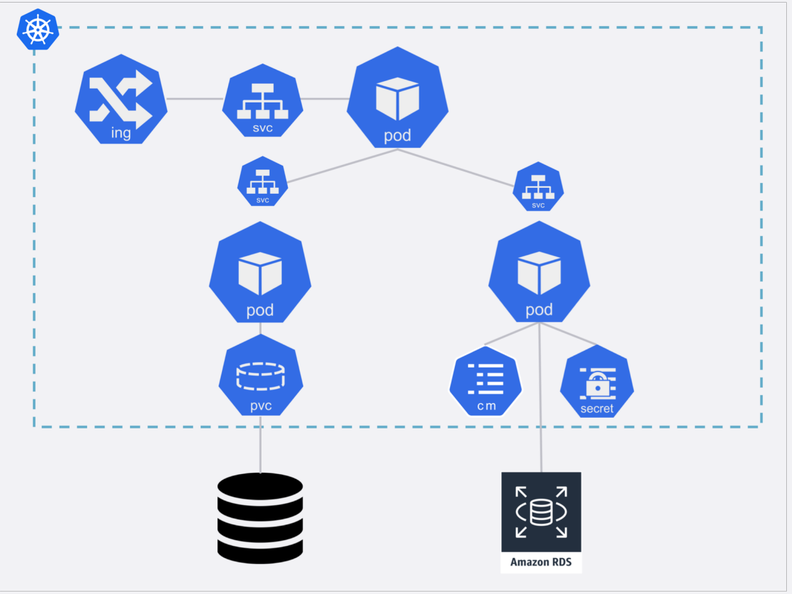
- Snapshot underlying volumes (crash-consistent)
- Application-level tools
  (app-consistent)
- A combination of both

Managed services outside K8s (self-hosted or cloud)

- Application-level tools
- Managed Service APIs

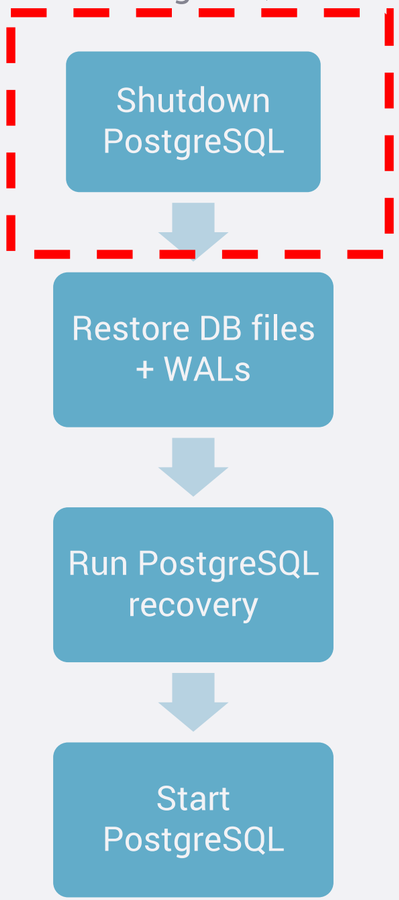
implementation
workflow orchestration

- Application requirements
  - Ordering across microservices
  - Quiescing
  - Pre/Post steps
- Kubernetes/Container interactions
  - Getting access to application data and volumes
  - Shutting down/Starting services



implementation
orchestration example

Recovery Playbook for PostgreSQL

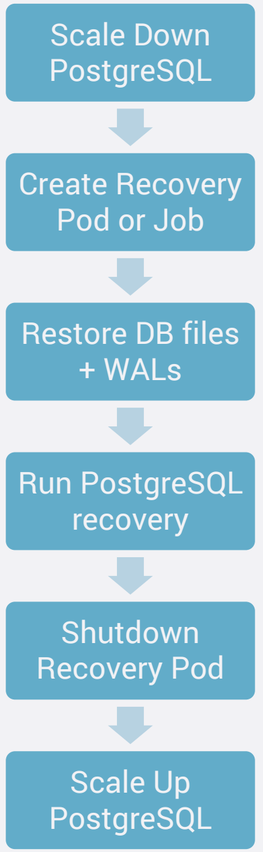


Pod will restart on PG shutdown

```
...
ENTRYPOINT ["docker-entrypoint.sh"]

EXPOSE 5432
CMD ["postgres"]
```

Orchestrating on Kubernetes



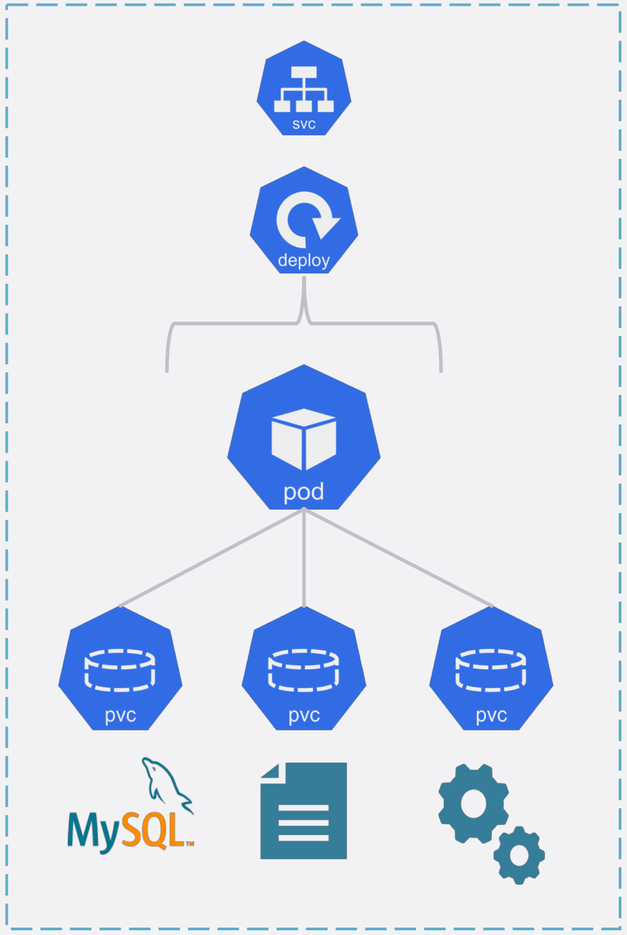
Use container image with
Postgres + Tools
Run custom commands
Attach PostgreSQL volumes (PVCs)

implementation
backup storage and format

- Where will backups be stored
  - Object Storage tends to be a good choice
- References to underlying data service snapshots
  - Durability
  - Portability
- Security and Encryption
  - Who has access to the data
  - Who can restore
  - Key management

demo and tools

demo
picture gallery demo app



Picture Gallery

- Deployment with 1 replica
- 3 Persistent Volumes
  - MySQL
  - Unstructured File Data
  - Config

kanister: Data management workflows in Kubernetes

- Describe data protection workflows using Kubernetes Custom Resources (CR)
- Primitives for data capture from (and into) a variety of data sources

- Workflow Orchestration

https://github.com/kanisterio

demo
backup workflow -> blueprint CR

Backup

- Discover PVCs
- Snapshot underlying Volumes
- Push Snapshot Info to Backup Storage

```yaml
apiVersion: cr.kanister.io/v1alpha1
kind: Blueprint
metadata:
  name: snapshot-blueprint
  namespace: demo
actions:
  backup:
    type: Deployment
    outputArtifacts:
      backupInfo:
      ...
    phases:
    - func: CreateVolumeSnapshot
      name: backupVolumes
```

demo
restore workflow -> blueprint CR

Restore

- Scale down application
- Delete existing PVCs
- Create new PVCs from snapshots
- Scale up application

```yaml
apiVersion: cr.kanister.io/v1alpha1
kind: Blueprint
metadata:
  name: snapshot-blueprint
  namespace: demo
actions:
  backup:
    ...
  restore:
    type: Deployment
    inputArtifactNames:
      - backupInfo
    phases:
    - func: ScaleWorkload
      name: shutdownPods
    - func: CreateVolumeFromSnapshot
      name: restoreVolumes
      args:
        snapshots: "{{ .ArtifactsIn.backupInfo }}"
    - func: ScaleWorkload
      name: bringupPods
```

tools

- Kanister
  - https://github.com/kanisterio/kanister
- Kasten K10
  - https://kasten.io
- Ark
  - https://github.com/heptio/ark
- ReShifter
  - https://github.com/mhausenblas/reshifter
- k8s-snapshots
  - https://github.com/miracle2k/k8s-snapshots
- Stash
  - https://github.com/appscode/stash
- Others
  - https://stateful.kubernetes.sh/#backup-and-restore

implementation
additional topics

- ## Backup Catalog
  Search, Discovery, Reporting, Auditing
- ## Scheduling and Retirement
- ## Restore Validation and Testing
- ## Integrating into CI/CD

Look for slides/recording soon from talk in the CI/CD track!

thank you

Questions?

You can also find us at:
Booth S/E15
www.kasten.io

@kastenhq @vaibhavkamra @deepikadixit

Image is the cover art from Better Together, a Jack Johnson song