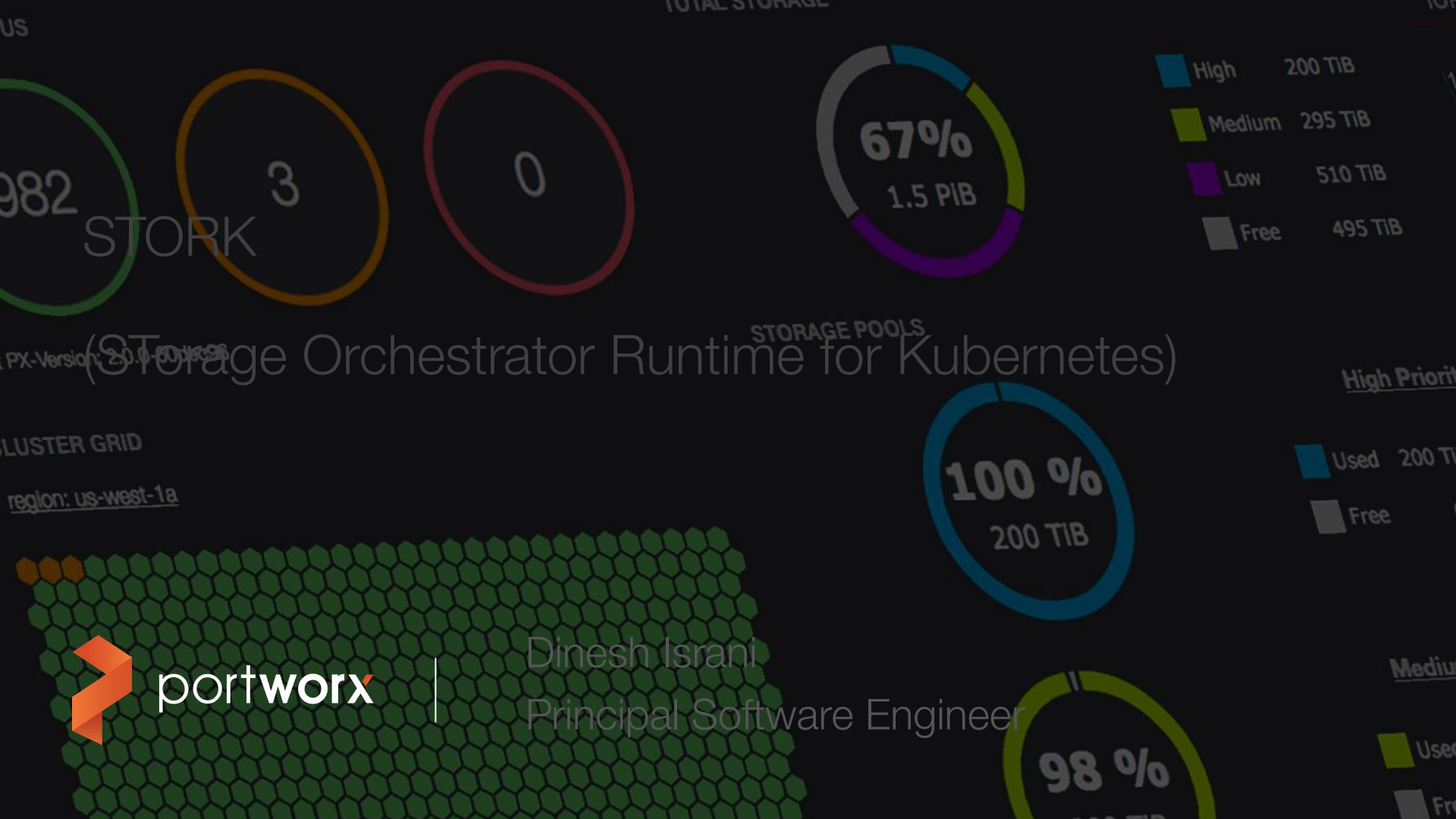


STORK

(STorage Orchestrator Runtime for Kubernetes)

portworx

Dinesh Israni
Principal Software Engineer

Agenda

- Introduction
- Motivation for Stork
- Scheduling stateful services efficiently
- Storage Health Monitoring
- Disaster Recovery (Snapshots, Backups, Migration)
- Q&A

Introduction

- Started and maintained by Portworx
- Open source: https://github.com/libopenstorage/stork
- Apache 2.0 License
- Started in November 2017, v1.0 GA in January 2018
- 23 releases made, next release (v2.3.0) scheduled for end of July

Some adopters

Motivation

- Help run stateful applications more efficiently on Kubernetes
  - Provide Hyper-convergence
  - Advanced health monitoring of stateful apps
- Manage lifecycle of stateful applications
  - Application consistent snapshots
  - Migrate applications between clusters
  - Backup Data + K8s resources
- Plugin model, can be extended to work with any storage driver

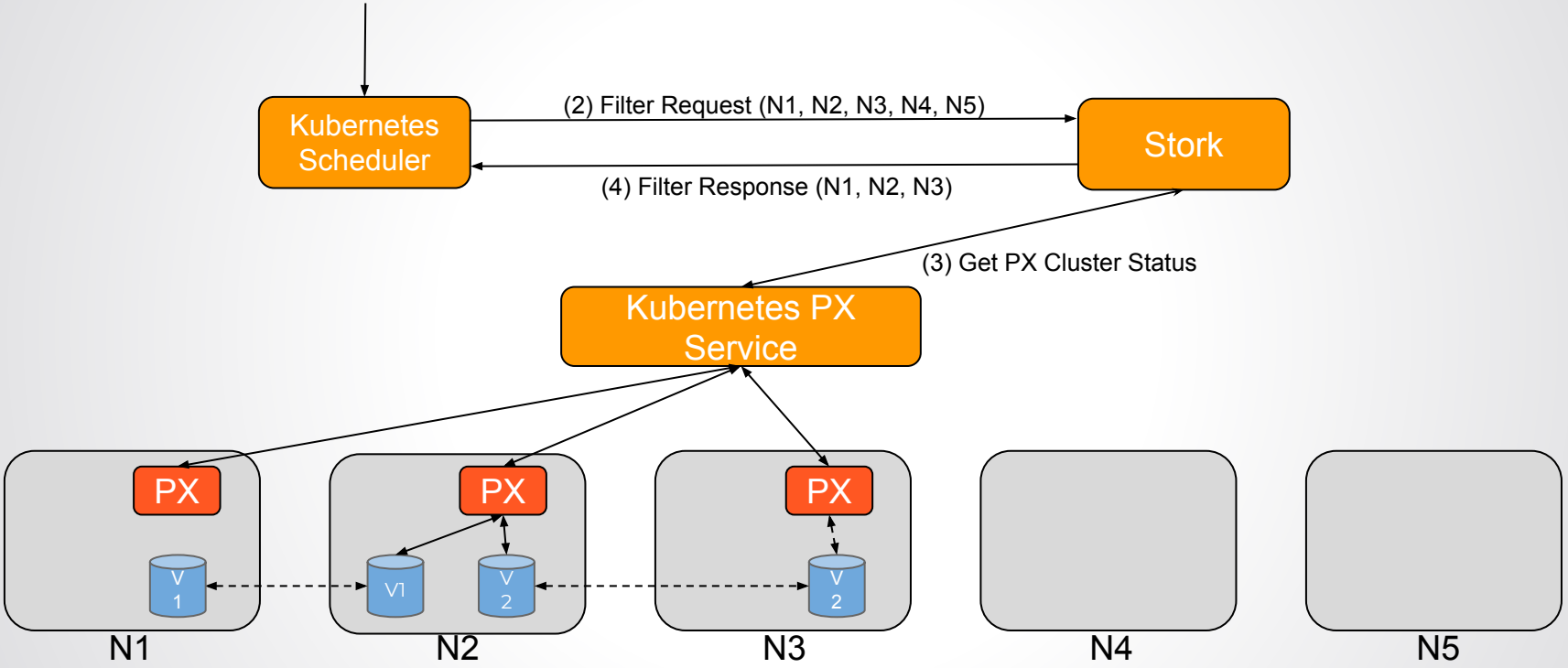
Scheduling stateful services efficiently

- How do you start pods close to where data is located?
- Wide use of labels and affinity rules
  - Doesn’t scale
  - Doesn’t work with stateful sets
  - Error prone

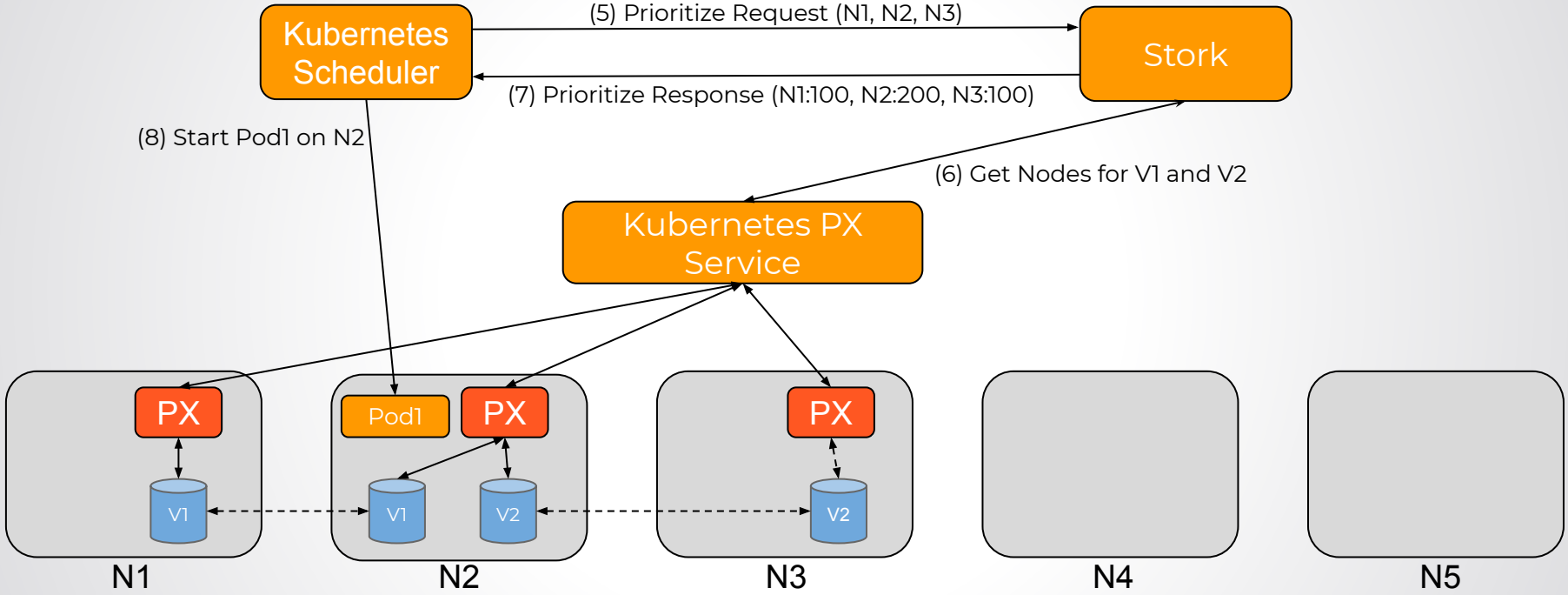
Solution: How to schedule stateful services

- Use scheduler extenders
- Kubernetes allows extending the default scheduler
- Can be used to
  - “filter” out nodes where storage isn’t available
  - “prioritize” nodes where data is local
- Simple to use
  - Either configure default scheduler with extender
  - Or, start new instance of scheduler and use in your apps
- Also have support for “initializer” to automatically set scheduler name for applications, but that got deprecated in K8s 1.14

Scheduling stateful services efficiently



Scheduling stateful services efficiently



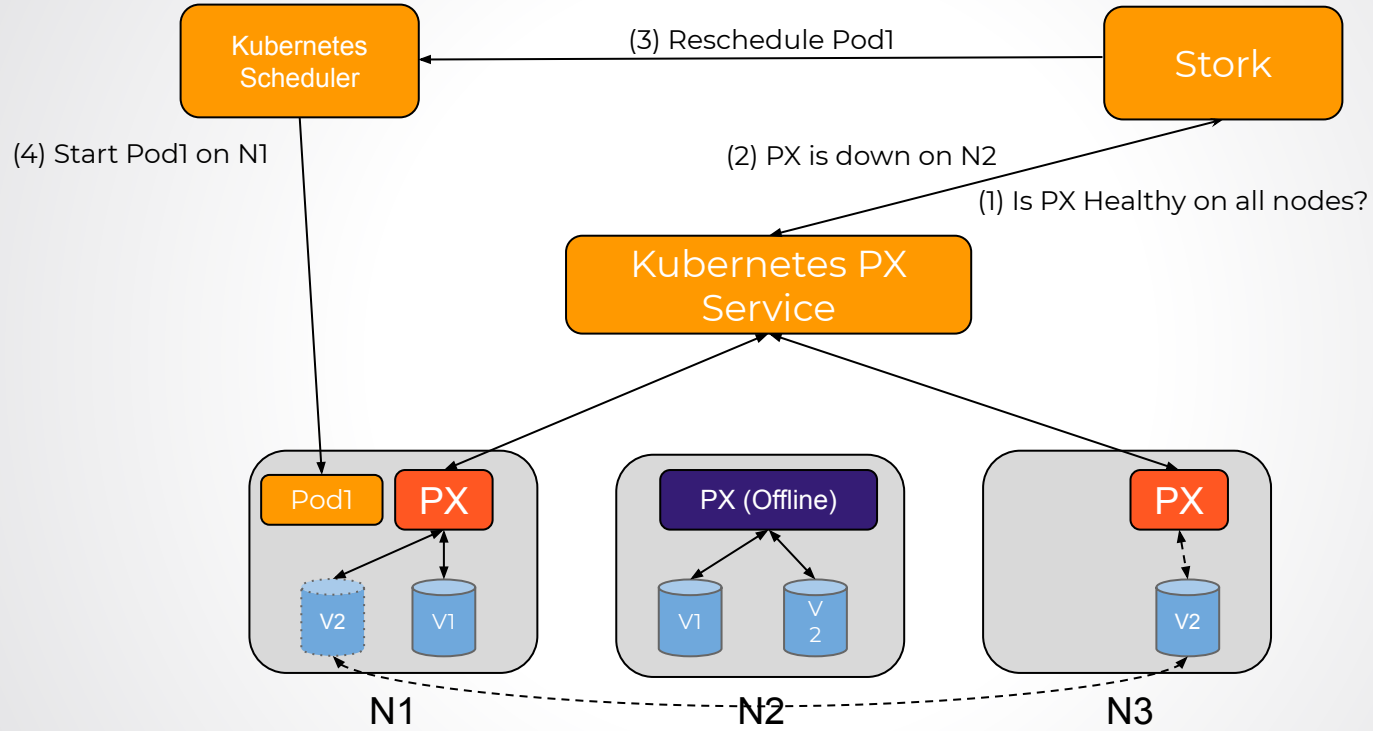
Challenges with Storage Health monitoring

- All good when everything is online
- Dealing with failures is difficult, especially with state
- What if storage driver goes offline on a node?
  - Storage degradation
  - Software bugs/crashes
- What happens to pods on that node?
  - Kubelet is still running
  - Pods will get stuck, app will not respond
  - Depending on the app, the health check might not fail either
- Stateful sets have a completely different behavior
  - They don’t reschedule pods even if kubelet stops responding on nodes
- Usually requires manual intervention

Storage Health Monitoring

- Monitors the health of storage driver on all nodes
- Storage driver offline?
  - Reschedule pods using storage driver
- Rescheduled on another node with volume replica
  - Continue with local disk performance
- Without this, pods will get stuck in Pending, or not able to access storage
- For stateful sets this also deals with scenarios where kubelet reports offline on a node

Storage Health Monitoring



Disaster Recovery - Snapshots & Application Backups

- Need a way to manage lifecycle of storage natively in Kubernetes
- There was no native support for snapshotting PVCs
- Added support for Snapshots (based on Kubernetes Incubator project)
- Also works over a group of PVCs / Volumes for distributed apps using GroupVolumeSnapshot CRD
- In v2.3.0
  - Backup data + application resources to an objectstore
  - In case of disaster, point new cluster to BackupLocation and restore the application
- Supports
  - Any S3 compliant objectstore
  - AzureBlob
  - GoogleCloudStorage

Disaster Recovery - Multi-Cloud / Multi-Cluster Migration

1. **Augmentation**: we are out capacity and want to move *select* applications and data to a second cluster.

2. **Blue-Green**: a new version of the storage driver is released, and we want to qualify with *all* applications and data.
(Also works for a new version of Kubernetes.)

3. **Dev/Test**: a bug in production needs to be reproduced off-cluster. We want to move *just* that app and its data.

Disaster Recovery - Multi-Cloud / Multi-Cluster Migration

- First pair 2 (or more) clusters
  - Pairs storage as well as Kubernetes across clusters
  - Can be any type of Kubernetes cluster (Vanilla K8s on-prem, GKE, AKS, EKS, IKS, OCP, etc)
- Then start migration across clusters
  - Specify namespace and labels to select which applications to migrate
- First migrates all the data then migrates K8s resources

Disaster Recovery - Schedules

- All operations can be scheduled
- SchedulePolicy CR can be used to specify when to trigger actions
  - Periodic, Daily, Weekly, Monthly
- Schedule CRs can then be created referring to the SchedulePolicy
  - VolumeSnapshotSchedule
  - MigrationSchedule
  - ApplicationBackupSchedule

Disaster Recovery - Application consistent operations

- Quiesce or flush applications before operation using pre/post execution rules
- Rules are defined in a CustomResource (CR) and referred to in Snapshot / Migration / ApplicationBackup objects
- Rules can be run either in the background or foreground while the operation is being executed
- Eg for mysql:
  - Flush tables and take a lock on the tables before taking snapshot
  - Needs to run in the background so that database lock is held
- Eg for Cassandra:
  - Flush all data from memory before taking snapshot
  - Needs to run in the foreground (ie before triggering the operation)

storkctl - Tool to view and manage CRs

```
$ storkctl create snap -n mysql -p mysql-data mysql-snapshot
Snapshot mysql-snapshot created successfully

$ storkctl get snap -n mysql
NAME             PVC         STATUS   CREATED                COMPLETED              TYPE
mysql-snapshot   mysql-data  Ready    09 Jul 19 02:15 UTC    09 Jul 19 02:15 UTC    local

$ storkctl get clusterpair --all-namespaces
NAMESPACE   NAME           STORAGE-STATUS   SCHEDULER-STATUS   CREATED
mysql       remotecluster  Ready            Ready              09 Jul 19 01:55 UTC

$ storkctl get migration -n mysql
NAME           CLUSTERPAIR    STAGE   STATUS      VOLUMES   RESOURCES   CREATED                ELAPSED
mysqlmigration remotecluster  Final   Successful  1/1       8/8         09 Jul 19 02:04 UTC    1m16s
```

Learn More

- Github: https://github.com/libopenstorage/stork
  - Welcome contributions for drivers and features
- Blogs:
  - https://portworx.com/stork-storage-orchestration-kubernetes/
  - https://portworx.com/free-compute-capacity-across-kubernetes-clusters-migrating-stateful-applications/
- Contact Info:
  - Email: disrani@portworx.com
  - Github: disrani-px
  - LinkedIn: https://www.linkedin.com/in/dineshisrani/
- Slack: http://portworx.slack.com

Q&A