Plan to Fail:
A good Captain Doesn’t Sail Without Life Rafts

Carlisia Campos, Steven Wong
VMware
Abstract

Hidden slide during presentation – included for those finding deck online later

Historically, formal disaster recovery (DR) plans were only feasible for large enterprises. They could afford to allocate time, resources and the cost of duplicating a datacenter infrastructure.

With the popularity of public cloud and cloud native technologies, the cost and complexity of DR planning has been significantly reduced. This means every company, large and small, can engage in business continuity planning. Why is this important? These are some of the reasons:

- Machines and software fail
- People make mistakes
- Hackers prey on the vulnerable
- Weather, fire, terrorism, more...
- You lose customers when there are outages and data loss
- Legal standards often require data retention

This talk will focus on:
- Items that need to be backup and why - some might surprise you
- Why you need selective restore capability
- Existing tooling to simplify and automate a DR strategy
Presenters

Carlisia Campos
San Diego
Senior member of Technical Staff, VMware
Carlisia is a maintainer of the open source project Velero, a cloud native disaster recovery and data migration tool for Kubernetes workloads.
GitHub: @carlisia

Steven Wong
Los Angeles
Open Source Community Relations Engineer, VMware
Active in Kubernetes community since 2015 – storage, IoT+Edge, running K8s on VMware infrastructure. Former engineer and architect of Avamar and other backup products.
GitHub: @cantbewong
Agenda

Why you need a recovery plan

Elements of a DR plan

Items you need to backup and why

Existing tools to implement backups and recovery

Demo
Why do you need a recovery plan?

Machines and software fail

People make mistakes

Hackers prey on the vulnerable

Weather, fire, terrorism, crime, earthquake, more...

Customers have alternatives - *Customer retention is costly, but customer re-acquisition is devastatingly expensive*

Legal standards often require data retention and impose a duty of reasonable care to protect against physical and financial harm
Elements of a Disaster Recovery Plan

All these work together

- backups
  possibly replicated to alternate sites
- replacements
  hardware or available cloud capacity
- preparation
  A planned process
- people
  people who can carry out the process
- training
  recurring training, with recovery tests
Not enough to just make a plan – record it as a living document

**Runbooks** documenting recovery procedures should be tested and retained offline – pre-installed on tablets or even printed
All *native* Kubernetes objects are stored on etcd. Periodically backing up the etcd cluster data is important to recover Kubernetes clusters under disaster scenarios, such as losing all master nodes.

etcd is also sometimes used to hold state for network plugins, CRDs, and other essential components.

BUT... Some critical state is held outside etcd.
Critical items outside etcd

protection and recovery plan needed:

• Persistent volumes
• Certificate and key pairs, Certificate Authority
• ServiceAccount signing
• LDAP or other authentication details
• State associated with any CRDs and CNI plugins not using etcd
• Network resources (configuration allowing recreation of DNS records, IP and subnet assignments, switch, firewall, routing, load balancing, proxies, etc.)
• Cloud provider specific account and configuration data
• Credentials for underlying infrastructure (access keys, tokens, passwords, etc.)
Redundancy alone is not enough

Helps reduce some types of outages… but

Bad updates, getting hacked, software bugs, or human error can simply replicate problems across redundant copies.

Better than a “mirror”:
Periodic backups of critical components to resilient storage
Container Images, Helm charts, other binaries and installables

Kubernetes workloads are based on container images.

You want the source and content of these images to be trustworthy. This will almost always mean that you will host a local container image repository.

If a repository is lost, repulling images from the public Internet can be time consuming and present security issues. If you use image signing, signatures would need to be reapplied.

You may wish to consider a recovery solution for your registries.

A registry solution supporting redundancy and image replication can be a good building block for a recovery plan.
some Kubernetes open source backup options
for Kubernetes state backup, restore, migrate via etcd or K8s API

<table>
<thead>
<tr>
<th>project</th>
<th>atomic or selective</th>
<th>contributors</th>
<th>persistent volume protection</th>
</tr>
</thead>
<tbody>
<tr>
<td>etcd native</td>
<td>atomic</td>
<td>498</td>
<td>no</td>
</tr>
<tr>
<td>Velero</td>
<td>selective</td>
<td>92</td>
<td>yes</td>
</tr>
<tr>
<td>ReShifter</td>
<td>atomic</td>
<td>6</td>
<td>no</td>
</tr>
<tr>
<td>kaptaind</td>
<td>atomic</td>
<td>1</td>
<td>no</td>
</tr>
</tbody>
</table>

selective means back/restore based on K8s namespaces, label selectors, etc
some Kubernetes **stateful app** open source backup options
via persistent volume and/or in container backup

<table>
<thead>
<tr>
<th>project</th>
<th>atomic or selective</th>
<th>project contributors</th>
<th>persistent volume protection</th>
<th>notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Velero</td>
<td>selective</td>
<td>96</td>
<td>yes</td>
<td>Restic optional</td>
</tr>
<tr>
<td>Stash</td>
<td>na</td>
<td>19</td>
<td>yes</td>
<td>Restic mandatory</td>
</tr>
<tr>
<td>K8up</td>
<td>na</td>
<td>7</td>
<td>yes</td>
<td>OpenShift only?</td>
</tr>
<tr>
<td>KubeMove</td>
<td>na</td>
<td>2</td>
<td>yes</td>
<td>not backup - for one time migration</td>
</tr>
</tbody>
</table>

we have carried over Velero because it offers both K8s state backup and app backup
### Use cases for Kubernetes backup solutions

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Disaster Recovery</strong></td>
<td>Restore clusters or applications after failures</td>
</tr>
<tr>
<td><strong>Data Archival</strong></td>
<td>Restore data that becomes lost or corrupt</td>
</tr>
<tr>
<td></td>
<td>Retire old data from expensive primary storage while retaining for compliance or future analytics</td>
</tr>
<tr>
<td><strong>Migration</strong></td>
<td>Migrate Kubernetes clusters or applications</td>
</tr>
</tbody>
</table>
Backup + Restore Demo

Photo by Dietmar Becker on Unsplash
Thank You
Q&A
Contacts
This deck: link tbd

- Kubernetes Velero Slack channel: [https://kubernetes.slack.com/messages/velero](https://kubernetes.slack.com/messages/velero)
- Velero open source project Community:
  - Join: [https://groups.google.com/forum/#!forum/projectvelero](https://groups.google.com/forum/#!forum/projectvelero)
  - Zoom meetings every 1st and 3rd Tuesday and recorded to YouTube channel
  - [github.com/heptio/velero-community](https://github.com/heptio/velero-community)

Carlisia Campos
@carlisia

Steven Wong
@cantbewong