mojaloop

Versioning

PI 11 - July 2020

Lewis Daly, Matt de Haast, Samuel Kummary

PI-10 Overview

1. PI 10 Objectives
2. Background: What have we done so far?
3. Switching to Semantic Versioning + LTS
4. `k8s-patcher`, applying security patches
5. Zero Downtime POC
6. Next Steps for the Versioning Stream

PI-10 Objective

Propose Versioning standards for Mojaloop OSS releases focusing on *implementation*, *deployments* that include details on OSS support for LTS releases, branching & upgrade strategies.

More specifically, we will do this with a **Zero Downtime Deployment Proof of Concept**

Semantic Versioning + LTS

mojaloop

Semantic Versioning in Mojaloop

Current standard, up to PI-10

1. The current versioning system is inspired by the Semantic Versioning numbering system for releases. However, this is customized to depict the timelines of the Mojaloop project, based on the Program Increment (PI) & Sprint numbers

2. *For example, the release number v5.1.0 implies that this release was the first one made during a Sprint 5.1, where Sprint 5.1 is the first Sprint in PI-5. So for a version vX.Y.Z, X.Y is the Sprint number where X is the PI number and Z represents the number of release for this specific repository. Example v4.4.4 implies that the current release is the fourth of four releases made in Sprint 4.4 (of PI-4)*

Semantic Versioning in Mojaloop

Standard for PI-11 and beyond

1. Move to a versioning system that's closely aligned with Semantic versioning (remove PI/Sprint dependency).

2. *At a high-level, we will still follow the v**X.Y.Z** format, but X represents 'Major' version, Y represents 'Minor' version and Z represents 'patch' version. Minor fixes, patches affect increments to 'Z', whereas non-breaking functionality changes affect changes to 'Y'; breaking changes affect the 'X' version.*

3. Along with these, suffixes such as "-snapshot", "-patch", "-hotfix" are used as relevant and on need basis.

4. So starting with 11.0.0 (*primarily for Helm, but for individual services as well*) for PI-11, the proposal is to move to pure SemVer. For PI10 however, the same versioning system outlined below was used.

Current Mojaloop Version

“Mojaloop Version”

Format: **x.y**: MAJOR.MINOR (An optional HOTFIX ‘z’ in x.y.z can be used for internal purposes, not broadcasted publicly)

Kicking-off usage of current Mojaloop version as **v1.1** which includes

1. APIs
   - FSPIOP API **v1.1**
   - Settlement API **v1.1**
   - Admin API **v1.0**
2. Helm **v10.4.0**
   - Individual services' versions
   - Monitoring components versions
   - Default config (and custom config privately for AWS environments)
3. Internal Schemas
   - DB Schema **v1.0**
   - Internal messaging version **v1.0**

Notes:

1. ML Version **v1.0** had everything same as ML v1.1 except for FSPIOP API **v1.0** and Helm **v10.1.0**
2. Read the full proposal here.

LTS for Mojaloop releases PI-11

Long term Support
Mojaloop has been evolving rapidly and Long term support wasn’t discussed so far for Open source (Helm) releases. However, the pattern followed so far was to support the latest available Helm release as requested.

Current LTS proposal for Mojaloop

1. For **PI-11**: Support the latest Helm release at the time (explicitly)
2. For **PI-12**: Support the last Helm release in the last PI (until the current PI’s last helm release)
3. For **2021** (PI-13 and onwards): Support every Helm release for a period of **3 months**
4. For **2022**: Support every Helm release for a period of **6 months**
5. Starting **2023**: Support every Helm release for a period of **1 year**

Mojaloop Operator (k8s)

k8s-patcher

How do we apply docker image and node module security patches to running deployments?

Security Patches & Images

1. Investigating best practices around rebuilding docker images with security patches, and how to apply them in running Kubernetes clusters

2. Images to be with a new security patch suffix:
    a. start from mojaloop/ml-api-adapter:v10.4.1
    b. publish a new version: mojaloop/ml-api-adapter:v10.4.1.1-patch when we apply a security patch (change of node_modules only)

3. Develop a new service that runs inside a deployment (k8s-patcher)
    a. Looks for updates to v10.4.1 (i.e. images with tag v10.4.1.X-patch)
    b. Provide prompts/notifications to Operators about availability of patched images
    c. Automatically (or manually) replaces pods with patched images

k8s-patcher

- Add options for deployments to deploy a Kubernetes Operator
- Labels on deployments used to determine which to watch
- Periodically poll deployments and check against registery for changes
- Notify Operator about updates

Demo time!

Zero Downtime POC

mojaloop

Zero Downtime Deployment Proof of Concept

Why?

- Demonstrate an approach for zero-downtime-deployments within Mojaloop OSS
- Learn Lessons to
  - Improve OSS Development practices
    - e.g. branching, release notes, handling breaking changes
  - Recommend Operator best practices

An Example Breaking Change

```
POST /transfers HTTP/1.1
Accept: application/vnd.interoperability.transfers+json;version=2
Date: Tue, 15 Nov 2020 08:12:31 GMT
... other headers...
Body:
{
  "transferId": "1234",
  "quoteId": "9876", <--- This is a new required field
  "payeeFsp": "dfspb",
  "payerFsp": "dfspa",
  "amount": "100.00",
  "ilpPacket": "XXXXX",
  "condition": "XXXXX",
  "expiration": "2020-05-24T08:38:08.699-04:00"
}
```

...would be 4x non-breaking helm releases

```
# apply the db schema changes: optional `quoteId` column
helm upgrade ml_release mojaloop/mojaloop --version v1.1

# add support for v2.0 API
helm upgrade ml_release mojaloop/mojaloop --version v1.2
# TODO: open up the v2.0 in as a preview in API-Gateway

# TODO:drain v1.X connections
# drop support for v1.X API
helm upgrade ml_release mojaloop/mojaloop --version v2.0

# apply final db schema changes: required `quoteId` column
helm upgrade ml_release mojaloop/mojaloop --version v2.1
```

...each of which has a Deployment like so:

```
apiVersion: apps/v1
kind: Deployment
metadata:
 name: mojaloop/central-ledger
 namespace: default
spec:
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxUnavailable: 25%
      maxSurge: 1
...
```

Caveats

Caveats

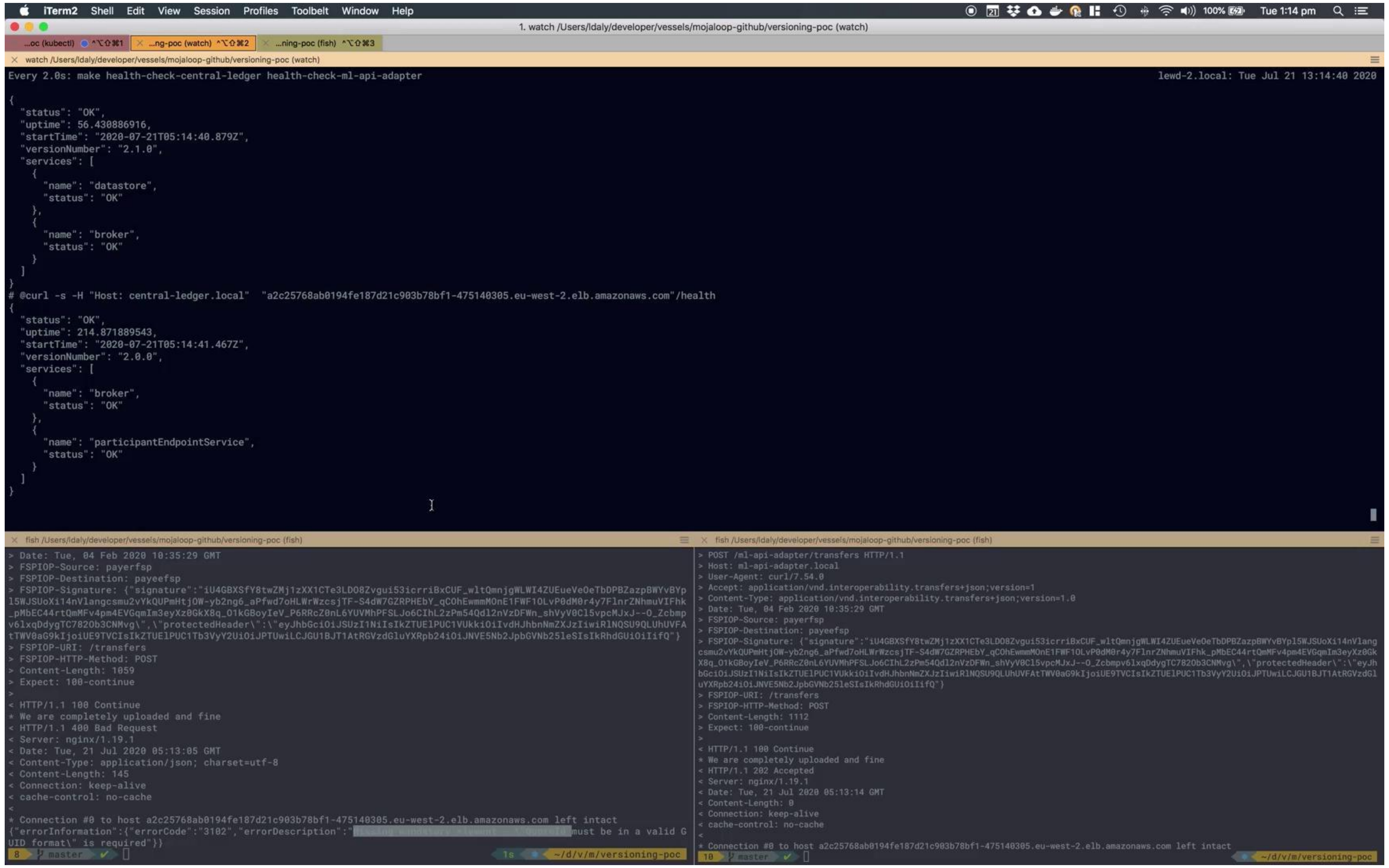
- This is a toy example

- No API Gateway
  - Instead we rely on Kubernetes Services only
  - No Resource-based version negotiation implemented

- We drop support for Mojaloop API v1.0, but operators might wish to keep support for much longer

It’s demo time

Demo Code Available: https://github.com/vessels-tech/versioning-poc


iTerm2 Shell Edit View Session Profiles Toolbelt Window Help
Tue 1:14 pm
1. watch /Users/ldaly/developer/vessels/mojaloop-github/versioning-poc (watch)
...oc (kubectl) ^⌥⇧⌘1
...ng-poc (watch) ^⌥⇧⌘2
...ning-poc (fish) ^⌥⇧⌘3
watch /Users/ldaly/developer/vessels/mojaloop-github/versioning-poc (watch)
fish /Users/ldaly/developer/vessels/mojaloop-github/versioning-poc (fish)
fish /Users/ldaly/developer/vessels/mojaloop-github/versioning-poc (fish)

```
Every 2.0s: make health-check-central-ledger health-check-ml-api-adapter          lewd-2.local: Tue Jul 21 13:14:40 2020

{
  "status": "OK",
  "uptime": 56.430886916,
  "startTime": "2020-07-21T05:14:40.879Z",
  "versionNumber": "2.1.0",
  "services": [
    {
      "name": "datastore",
      "status": "OK"
    },
    {
      "name": "broker",
      "status": "OK"
    }
  ]
}
# @curl -s -H "Host: central-ledger.local"  "a2c25768ab0194fe187d21c903b78bf1-475140305.eu-west-2.elb.amazonaws.com"/health
{
  "status": "OK",
  "uptime": 214.871889543,
  "startTime": "2020-07-21T05:14:41.467Z",
  "versionNumber": "2.0.0",
  "services": [
    {
      "name": "broker",
      "status": "OK"
    },
    {
      "name": "participantEndpointService",
      "status": "OK"
    }
  ]
}
```

```
> Date: Tue, 04 Feb 2020 10:35:29 GMT
> FSPIOP-Source: payerfsp
> FSPIOP-Destination: payeefsp
> FSPIOP-Signature: {"signature":"iU4GBXSfY8twZMj1zXX1CTe3LDO8Zvgui53icrriBxCUF_wltQmnjgWLWI4ZUEueVeOeTbDPBZazpBWYvBYp
l5WJSUoXi14nVlangcsmu2vYkQUPmHtjOW-yb2ng6_aPfwd7oHLWrWzcsjTF-S4dW7GZRPHEbY_qCOhEwmmMOnE1FWF1OLvP0dM0r4y7FlnrZNhmuVIFhk
_pMbEC44rtQmMFv4pm4EVGqmIm3eyXz0GkX8q_O1kGBoyIeV_P6RRcZ0nL6YUVMhPFSLJo6CIhL2zPm54Qdl2nVzDFWn_shVyV0Cl5vpcMJxJ--O_Zcbmp
v6lxqDdygTC782Ob3CNMvg\",\"protectedHeader\":\"eyJhbGciOiJSUzI1NiIsIkZTUElPUC1VUkkiOiIvdHJhbnNmZXJzIiwiRlNQSU9QLUhUVFA
tTWV0aG9kIjoiUE9TVCIsIkZTUElPUC1Tb3VyY2UiOiJPTUwiLCJGU1BJT1AtRGVzdGluYXRpb24iOiJNVE5Nb2JpbGVNb25leSIsIkRhdGUiOiIifQ"}
> FSPIOP-URI: /transfers
> FSPIOP-HTTP-Method: POST
> Content-Length: 1059
> Expect: 100-continue
>
< HTTP/1.1 100 Continue
* We are completely uploaded and fine
< HTTP/1.1 400 Bad Request
< Server: nginx/1.19.1
< Date: Tue, 21 Jul 2020 05:13:05 GMT
< Content-Type: application/json; charset=utf-8
< Content-Length: 145
< Connection: keep-alive
< cache-control: no-cache
<
* Connection #0 to host a2c25768ab0194fe187d21c903b78bf1-475140305.eu-west-2.elb.amazonaws.com left intact
{"errorInformation":{"errorCode":"3102","errorDescription":"Missing mandatory element - \"quoteId\" must be in a valid G
UID format\" is required"}}
 8  master ✔                                                                   1s  ~/d/v/m/versioning-poc
```

```
> POST /ml-api-adapter/transfers HTTP/1.1
> Host: ml-api-adapter.local
> User-Agent: curl/7.54.0
> Accept: application/vnd.interoperability.transfers+json;version=1
> Content-Type: application/vnd.interoperability.transfers+json;version=1.0
> Date: Tue, 04 Feb 2020 10:35:29 GMT
> FSPIOP-Source: payerfsp
> FSPIOP-Destination: payeefsp
> FSPIOP-Signature: {"signature":"iU4GBXSfY8twZMj1zXX1CTe3LDO8Zvgui53icrriBxCUF_wltQmnjgWLWI4ZUEueVeOeTbDPBZazpBWYvBYpl5WJSUoXi14nVlang
csmu2vYkQUPmHtjOW-yb2ng6_aPfwd7oHLWrWzcsjTF-S4dW7GZRPHEbY_qCOhEwmmMOnE1FWF1OLvP0dM0r4y7FlnrZNhmuVIFhk_pMbEC44rtQmMFv4pm4EVGqmIm3eyXz0Gk
X8q_O1kGBoyIeV_P6RRcZ0nL6YUVMhPFSLJo6CIhL2zPm54Qdl2nVzDFWn_shVyV0Cl5vpcMJxJ--O_Zcbmpv6lxqDdygTC782Ob3CNMvg\",\"protectedHeader\":\"eyJh
bGciOiJSUzI1NiIsIkZTUElPUC1VUkkiOiIvdHJhbnNmZXJzIiwiRlNQSU9QLUhUVFAtTWV0aG9kIjoiUE9TVCIsIkZTUElPUC1Tb3VyY2UiOiJPTUwiLCJGU1BJT1AtRGVzdGl
uYXRpb24iOiJNVE5Nb2JpbGVNb25leSIsIkRhdGUiOiIifQ"}
> FSPIOP-URI: /transfers
> FSPIOP-HTTP-Method: POST
> Content-Length: 1112
> Expect: 100-continue
>
< HTTP/1.1 100 Continue
* We are completely uploaded and fine
< HTTP/1.1 202 Accepted
< Server: nginx/1.19.1
< Date: Tue, 21 Jul 2020 05:13:14 GMT
< Content-Length: 0
< Connection: keep-alive
< cache-control: no-cache
<
* Connection #0 to host a2c25768ab0194fe187d21c903b78bf1-475140305.eu-west-2.elb.amazonaws.com left intact
 10  master ✔                                                                  ~/d/v/m/versioning-poc
```

Lessons Learned - OSS Developers

- Breaking changes (API, Database Schema, Config) must be decomposed into many non-breaking changes

- Adjacent service versions (e.g. `central-ledger:v2.0.0` and `central-ledger:v2.1.0`) might be deployed alongside each other, and therefore must be compatible and tolerant

- We can improve our processes for automating **releases** and **release notes**
  - Both on individual services and on helm releases
  - There are existing frameworks + tools that do this: such as *conventional commits*

- We need to put more thought into our **database schema changes**
  - At the moment, rolling back the application after a schema change (without rolling back the schema) causes the application to crash

Lessons Learned - Operators

- Skipping intermediate versions without downtime looks rather tricky

- Every major API change will require its own upgrade strategy

Versioning Stream in PI-11

- Continue work on k8s-patcher
- Improve our development pipelines with automated releases and release notes
- Improve our helm charts to deploy without downtime by default
- Expand our continuous delivery capability and QA Testing