DevOps Helm Charts

June 24, 2025

Table of Contents

Get Started

Welcome

This is the documentation for using Helm to deploy the Ping Identity container images. This single chart can be used to deploy any of the available Ping Identity products in a Kubernetes environment.

DevOps Resources

Ping DevOps	Docker Images	GitHub Repos	Community

Prerequisites

- Kubernetes 1.16+
- Helm 3
- Ping Identity DevOps user/key

Adding the Helm Repo

```
helm repo add pingidentity https://docs.pingidentity.com/helm/
```

Removing the Repo

```
helm repo rm pingidentity
```

Getting Started

Helm is a package deployment tool for Kubernetes. It can be used with Ping DevOps to deploy all the components of the solution with a simple command.

Prerequisites

- Kubernetes Cluster
- Helm 3
- Ping Identity DevOps user/key

> **Note**
>
> Ping Helm charts support OpenShift. See OpenShift Configuration to learn how to configure the values.yaml file to do so.

Creating Ping DevOps Secret

The charts use a secret called `devops-secret` to obtain an evaluation license for running images.

- Eval License — Use your `PING_IDENTITY_DEVOPS_USER/PING_IDENTITY_DEVOPS_KEY` credentials along with your `PING_IDENTITY_ACCEPT_EULA` setting.
- For more information on obtaining credentials, click here.
- For more information on using the `pingctl` utility, click here.

```
pingctl k8s generate devops-secret | kubectl apply -f -
```

Installing Helm 3

Ensure that you have Helm 3 installed.

- Installing on macOS (or Linux with Brew):

  `brew install helm`

- Installing on other OS:

  https://helm.sh/docs/intro/install/

Adding Helm Ping DevOps Repo

```
helm repo add pingidentity https://developer.pingidentity.com/helm/
```

Listing Ping DevOps Charts

```
helm search repo pingidentity
```

Updating Local Machine with the Latest Charts

```
helm repo update
```

Installing the Ping DevOps Chart

Install the `ping-devops` chart using the example below. In this case, it is installing a release called `pf`:

- PingFederate Admin instance
- PingFederate Engine instance

```
helm install pf pingidentity/ping-devops \
   --set pingfederate-admin.enabled=true \
   --set pingfederate-engine.enabled=true
```

or, if you have a `ping-devops-values.yaml` file:

```
# ping-devops-values.yaml
pingfederate-admin:
  enabled: true

pingfederate-engine:
enabled: true
```

```
helm install pf pingidentity/ping-devops \
-f ping-devops-values.yaml
```

Accessing Deployments

By default, the components of a release are prefixed with the release name. Continuing this example, everything will be prefixed with `pf`. Use `kubectl` to see the pods created.

View Kubernetes resources installed:

```
# get just pods
kubectl get pods --selector=app.kubernetes.io/instance=pf

# or get even more
kubectl get all --selector=app.kubernetes.io/instance=pf
```

View logs (from deployment):

```
kubectl logs deployment/pf-pingfederate-admin
```

Uninstalling Releases

To uninstall a release from Helm, use the following command (using `pf` as an example release):

```
helm uninstall pf
```

Contributing to the Ping Identity DevOps Program

Thanks for taking the time to help us improve our Helm chart!

You can contribute in various ways.

Reporting Bugs

Bugs are tracked as GitHub issues. You can report a bug by submitting an issue in the project's issue tracker. To help the maintainers understand and reproduce the problem, please try to provide detailed information, including:

- A clear and descriptive title.
- A description of what happened and what you expected to happen.
- An example with the exact steps needed to reproduce the problem. If relevant, sample code is helpful.

Please understand that bug reports are reviewed and prioritized internally, and we may not be able to address all bug reports or provide an estimated time for resolution.

Suggesting Enhancements

As with bugs, requests are tracked as GitHub issues. You can suggest an enhancement by submitting an issue in the project's issue tracker.

Please understand that enhancement requests are handled in the same way as bug reports, and we may not be able to address all enhancement requests or provide an estimated time for resolution.

> **Note**
>
> If you would rather not have your issue discussed in public, you can email bug reports or enhancement requests to devops_program@pingidentity.com.

Contributing Code Changes

Ping Identity does not accept third-party code submissions.

Value Configs

Introduction

The charts make heavy use of `Values` yaml files to pass configuration details to the Helm Charts. As defined by Helm Values Files, values are provided to the chart using the following mechanisms:

- `values.yaml` file in the chart
- Value files passed to Helm during install/upgrade with the `-f` flag
- Individual parameters passed with the `--set` flag

The list above is in order of specificity: `values.yaml` in the chart can be overridden with `-f` supplied files, which can in turn be overridden with the `--set` parameter.

The example below shows how values from the chart, a user-supplied `myconfig.yaml` file, and `--set` parameters are merged with each other to form merged values.

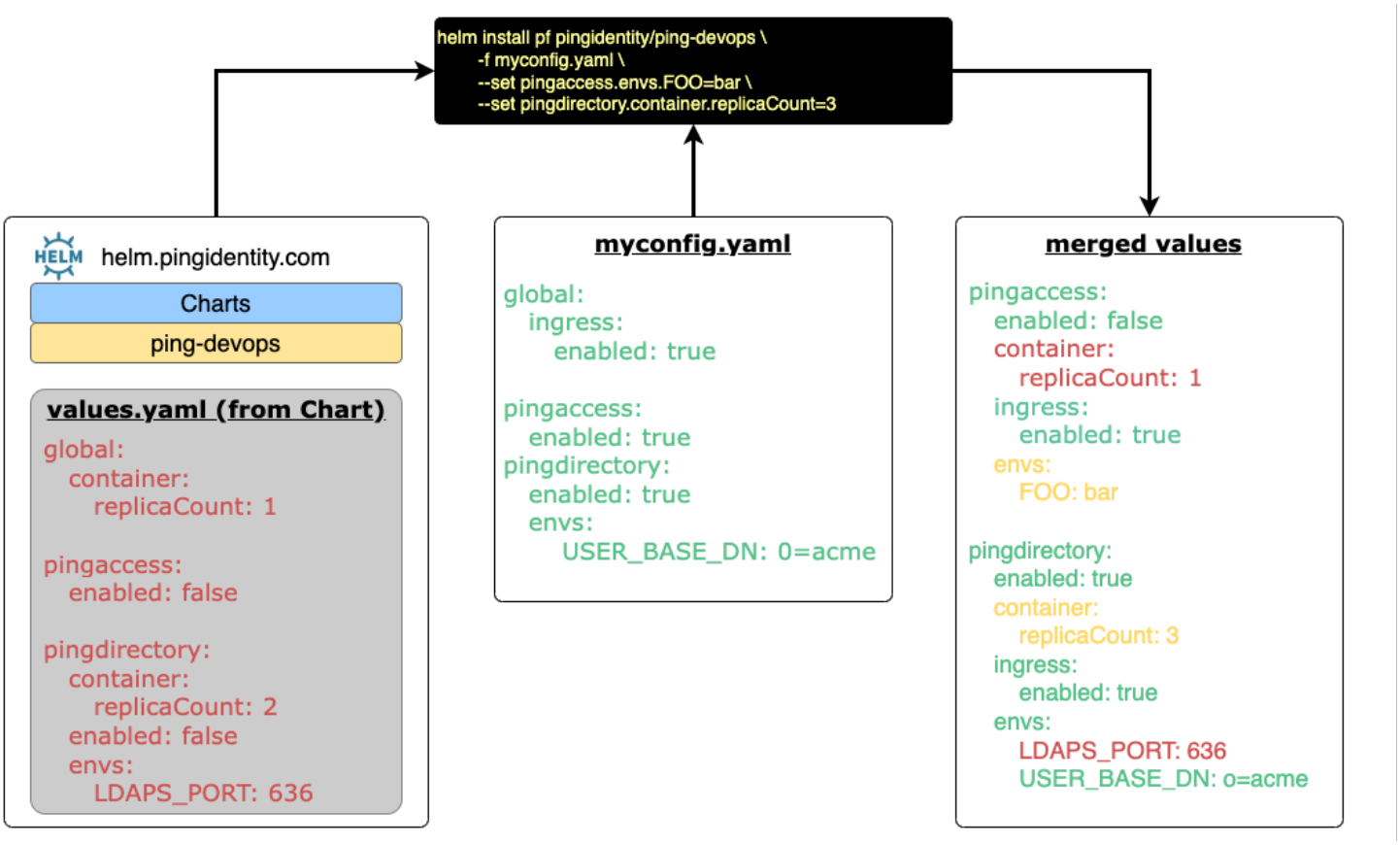

Chart Values

To see the values supplied by the chart, simply use the `helm show values` command to print them. This provides both the data as ell as context-sensitive comments to each section.

```
helm show values pingidentity/ping-devops
##############################################################
# Ping Identity DevOps values.yaml
##############################################################
# ...
```

You can also see all the available values in the helm-charts repository on GitHub.

The default values are broken up into two major sections:

- **global** — Represents the base set of values that will be provided to each product section unless it's overridden in that section.
- **product** — For every image/product, the values will be merged with the global settings and take precedence.

> **Warning**
>
> Values can only be modified when merged. They cannot be deleted. Also, if a value is set to the boolean true, and merged with a boolean of false, it will always be true.

Global Section

The `global:` section of the values contains configuration that is available to each product section. If a value is set in globals, that will be available to every product. This is very powerful, as you can turn on the ingress for every product by simply setting:

```
global:
  ingress:
    enabled: true
```

This would in essence set `ingress.enabled=true` for every product:

```
pingaccess:
  ingress:
    enabled: true

pingdirectory:
  ingress:
    enable: true
```

and so on.

It is much easier to set something in the `global:` section rather than repeat it for each product. To enable the ingress for only a few specific products, leave the default of `global.ingress.enabled=false` and set that value for those product sections.

Product Sections

Just like the `global:` values, each product can have the same values as well as many more that are specific to that product/image. In the following example, persistent volume configuration is provided for PingDirectory:

```
pingdirectory:
  persistentvolume:
    enabled: true
    volumes:
      - name: out-dir
        mountPath: /opt/out
        storage: 8Gi
        storageClassName:
```

global: Values

There is a top-level `global` value providing instructions on how to name all Kubernetes resources, so a deployer might deploy several releases under the same namespace.

addReleaseNameToResource

Provides global ability to add the Helm `.Release.Name` to Kubernetes resources.

Value	Description	Example: (Release.Name=acme, resource=pingdirectory)
prepend	Prepends Release.Name [DEFAULT]	acme-pingdirectory
append	Appends Release.Name	pingdirectory-acme
none	No use of Release.Name	pingdirectory

Container Configuration

Kubernetes Workload Controller resources are created depending on configuration values:

- Deployments
- StatefulSets

Global Section

Default yaml defined in the global `container` section:

```
global:
  container:
    replicaCount: 1
    resources:
      requests:
        cpu: 0
        memory: 0
      limits:
        cpu: 0
        memory: 0
    nodeSelector: {}
    tolerations: []
    affinity: {}
    terminationGracePeriodSeconds: 30
    envFrom: []
    lifecyle: {}
    probes:
      livenessProbe:
        exec:
          command:
            - /opt/liveness.sh
        initialDelaySeconds: 30
        periodSeconds: 30
        timeoutSeconds: 5
        successThreshold: 1
        failureThreshold: 4
      readinessProbe:
        exec:
          command:
            - /opt/readiness.sh
        initialDelaySeconds: 30
        periodSeconds: 5
        timeoutSeconds: 5
        successThreshold: 1
        failureThreshold: 4
      startupProbe:
         exec:
           command:
             - /opt/liveness.sh
         periodSeconds: 10
         timeoutSeconds: 5
         failureThreshold: 90
```

Probes Configuration

Kubernetes Probes defined in the `container:` section will be added to workloads (that is, Deployments/StatefulSets).

Fields used to configure probes can be found in the Kubernetes documentation.

External Image Configuration

Defines an external image for initContainer utilities.

Global Section

Default yaml defined in the global `externalImage` section:

```
global:
  externalImage:
    pingtoolkit: pingidentity/pingtoolkit:latest
```

External Image Parameters	Description
pingtoolkit	Registry, image and tag location for pingtoolkit. Used for primarily during init containers.

> **Note**
>
> If your Kubernetes cluster doesn't have access to an external Docker repository, you can download and save the `pingtoolkit` image to your local repo. Setting this to your local repo will cause the charts to use that image.

Image Configuration

Provides values to define kubernetes image information to deployments and statefulsets.

Global Section

Default image yaml defined in the global section:

```
global:
  image:
    repository: pingidentity
    name:        # Set in product section
    tag: 2307
    pullPolicy: Always
  imagePullSecrets: []  # As needed for authentication to private repositories
  # - name: myregkeysecretname
```

Product Section

Each product section specifies the name by default.

```
pingaccess-admin:
  image:
    name: pingaccess
```

To have images use a different repository and tag, use the following:

```
global:
  image:
    tag: edge
    repository: my.company.docker-repo.com
```

This snippet would result in pulling a PingAccess image from `my.company.docker-repo.com/pingaccess:edge`.

Ingress Configuration

Kubernetes Ingress resources are created depending on configuration values.

Global Section

Default yaml defined in the global `ingress` section, followed by definitions for each parameter:

```
global:
  ingress:
    enabled: false
    addReleaseNameToHost: subdomain
    defaultDomain: example.com
    defaultTlsSecret:
    annotations: {}
    spec: {}
```

Ingress Parameters	Description	Options	Default Value
enabled	Enables ingress definition.		false
addReleaseNameToHost	How `helm release-name` should be added to host.	prepend append subdomain none	subdomain
defaultDomain	Default DNS domain to use. Replaces the string "_defaultDomain_".		
defaultTlsSecret	Default TLS Secret to use. Replaces the string "_defaultTlsSecret_".		example.com
annotations	Annotations are used to provide configuration details to specific ingress controller types.	* see option for nginx ingress	{}
spec.ingressClassName	This value is replacing the `kubernetes.io/ingress.class` annotation. See this page for details.	name of the IngressClass resource	{}

Annotations example for nginx ingress:

```
annotations:
  nginx.ingress.kubernetes.io/backend-protocol: "HTTPS"
```

ingressClassName specification example for nginx ingress:

```
 spec:
   # Must match the name of the IngressClass resource
   ingressClassName: nginx-public
```

Product Section

Default yaml defined in the product `ingress` section, followed by definitions for each parameter:

```
ingress:
  hosts:
    - host: pingfederate-admin._defaultDomain_
      paths:
      - path: /
        backend:
          serviceName: admin
  tls:
    - secretName: _defaultTlsSecret_
      hosts:
```

Ingress Parameters	Description	Default Value
hosts	Array of hosts definitions	
hosts[].host	Full DNS name of host to use for external name. "_defaultDomain_" will be replaced with .defaultDomain	{product-name}._defaultDomain_
hosts[].paths	Array of paths to define for host	
hosts[].paths[].path	Path on external ingress	
hosts[].paths[].backend.serviceName	Name of the service to map to. This will result in the ingressPort on the server to be used.	
tls	Array of tls definitions	
tls[].secretName	Certificate secret to use	_defaultTlsSecret_
tls[].hosts	Array of specific hosts	

Example use of _defaultDomain_ and addReleaseNameToHost:

```
     helm ReleaseName = acme
        defaultDomain = example.com
 addReleaseNameToHost = subdomain
ingress.hosts[0].host = pingfed-admin._defaultDomain_

Resulting host will be: pingfed-admin.acme.example.com
                                      ^    ^^^^^^^
                                      |       |
                             ReleaseName    defaultDomain
```

Example Ingress Manifest

Below is an example product ingress for `pingfederate-admin` when deployed by Helm with a release-name of `acme`. It includes an ingress for the admin service (9999) using the default domain and tls secret, defined in the global section (if set).

```
kind: Ingress
metadata:
  annotations:
    ....
  spec:
    rules:
      - host: pingfederate-admin.acme.example.com
        http:
          paths:
          - backend:
              serviceName: acme-pingfederate-admin
              serviceName: admin
            path: /
      tls:
      - hosts:
      - pingfederate-admin.acme.example.com
      secretName: ""
```

License Configuration

Provides a secret used for obtaining evaluation licenses for Ping Identity products.

Global Section

Default yaml defined in the global `license` section, followed by definitions for each parameter:

```
global:
  license:
    secret:
      devOps: devops-secret
```

License Parameters	Description	Default Value
secret.devops	Secret containing PING_IDENTITY_DEVOPS_USER/KEY values.	devops-secret

> **Note**
>
> Use the `pingctl` command-line tool to create the devops-secret with your Ping Identity DevOps user and key.
> `pingctl kubernetes generate devops-secret | kubectl apply -f`

PrivateCert Configuration

Generates a private certificate (.crt and .key) based on the internal hostname of the service.

Global Section

> **Note**
>
> privateCert is currently only supported by PingAccess.

Default yaml defined in the global `privateCert` section. By default, certificates will not be generated. It is advised to NOT generate internal certs at the global level, as many services don't need a private cert on the internal service.

```
global:
############################################################
# Internal Certificates
#
# If set to true, then an internal certificate secret will
# be created along with mount of the certificate in
# /run/secrets/internal-cert (creates a tls.crt and tls.key)
#
# By default the Issuer of the cert will be the service name
# created by the Helm Chart.  Additionally, the ingress hosts,
# if enabled, will be added to the list of X509v3 Subject Alternative Name
#
# Use the additionalHosts and additionalIPs if additional custom
# names and ips are needed.
#
#      privateCert.generate: {true | false}
#      privateCert.additionalHosts: {optional array of hosts}
#      privateCert.additionalIPs: {optional array of IP Addresses}
############################################################
privateCert:
  generate: false
  additionalHosts: []
  additioanlIPs: []
```

Product Section

Generating an internal certificate is as simple as setting `privateCert.generate` to true.

Here's an example of generating an internal certificate for `pingaccess-engine`:

```
pingaccess-admin:
  privateCert:
    generate:true
```

This will ultimately create a secret named `{release-productname}-private-cert` containing a valid `tls.crt` and `tls.key`.

By default, the issuer of the cert will be the service name created by the Helm Chart. Additionally, the ingress hosts, if enabled, will be added to the list of `X509v3 Subject Alternative Name`.

The product image will then create an init container to generate a pkcs12 file that will be placed in `/run/secrets/private-keystore/keystore.env`, which will be mounted into the running container.

When the container's hooks are running, it will source the environment variables in this keystore.env. The default variables set are:

- `PRIVATE_KEYSTORE_PIN={base64 random pin}`
- `PRIVATE_KEYSTORE_TYPE=pkcs12`
- `PRIVATE_KEYSTORE={pkcs12 keystore}`

These environment variables are required in the `data.json.subst` file in order to use the generated privateCert. They can be used in any server-profile artifacts to be replaced when the images are started.

Service Configuration

Kubernetes Service resources are created depending on configuration values.

Product Section

Default yaml defined in the product `services` section. The example found in the `pingfederate-admin` section is:

```
services:
  admin:
    servicePort: 9999
    containerPort: 9999
    ingressPort: 443
    dataService: true
  clusterbind:
    servicePort: 7600
    containerPort: 7600
    clusterService: true
  clusterfail:
    servicePort: 7700
    containerPort: 7700
    clusterService: true
  clusterExternalDNSHostname:
```

Service Parameters	Description
services	Array of services
services[].{name}	Service Name. (i.e. https, ldap, admin, api)
services[].{name}.servicePort	External port of service
services[].{name}.containerPort	Port on target container
services[].{name}.ingressPort	Port on ingress container (if ingress is used)
services[].{name}.dataService	Adds to a ClusterIP service with single DNS/IP
services[].{name}.clusterService	Adds to a headless service with DNS request returning all IPs
services.clusterExternalDNSHostname	

The example above will create a container/service/ingress that looks like this:

Vault Configuration

The current Helm chart support is provided for Hashicorp Vault annotations and use of the Hashicorp injector. More information on Hashicorp Vault annotations can be found here.

> **Note**
>
> The PingIdentity DevOps images and Helm chart only support version 2 of the KV secrets engine API for Vault secrets. PingDirectory itself currently only supports KV version 1 for password storage schemes. Learn more in the Vault KV secrets engine documentation.

Vault Secret Values

An example vault values section looks like:

```
vault:
  enabled: true
  hashicorp:
    annotations:
      role: {hashicorp-vault-role}
    secretPrefix: {path to secret}
    secrets:
      {secret-name}:
        {secret-key | to-json}:
          path: /opt/in/some/location/secrets
          file: devops-secret.env
```

The `vault.hashicorp.secrets` is a map that specifies each `secret` to pull from the vault. And for each secret, a map specifies the `key` to pull with instructions of where to place the secret based on `path` and `file`.

License Parameters	Description	Default Value
secrets.{secret}	map of secret	devops-secret
secrets.{secret}.{key}	map of key	pingaccess.lic
secrets.{secret}.{key}.path	optional: location of secret. Defaults to vault.annotation.secret-volume-path	/opt/in/some/path
secrets.{secret}.{key}.file	required: file name secrets placed into	pingaccess.lic

Special Key Name (to-json)

There is a special key name that can be provided that will drop the raw secret into the container as its json representation with all the secret key names/values.

If dropped into the `SECRETS_DIR` (defaults to `/run/secrets`) directory, these files will be processed as:

- PROPERTY_FILE if the file ends in `.env`, or
- Separate files will be created for each key=value pair.

See the example below in this document for the transformation that occurs with the `devops-secret.env`.

Vault Annotations

Default yaml defined in the global `vault` section. The options of annotation names/values can be found at vault definitions.

For each of the annotations, the helm chart will automatically pre-pend the annotation with the hashicorp annotation prefix of `vault.hashicorp.com`. See example below.

```
global:
  vault:
    enabled: false
    hashicorp:
      annotations:
        agent-inject: true
        agent-init-first: true
        agent-pre-populate-only: true
        log-level: info
        preserve-secret-case: true
        role: k8s-default
        secret-volume-path: /run/secrets
```

The serviceAccount used by Vault will match the default serviceAccount for the workload.

Example

The following includes an example Hashicorp Vault secrets as well as a value values.yaml that make use of the secrets and an example of where secrets will be placed into the container.

Example: Hashicorp Vault secrets

```
SECRET:secrets/jsmith@example.com/jsmith-namespace/licenses

{
"pingaccess-6.2": "Product=PingAccess\nVersion=6.2...",
"pingdirectory-8.2": "Product=PingDirectory\nVersion=8.2...",
"pingfederate-10.2": "Product=PingFederate\nVersion=10.2..."
}

SECRET: secrets/jsmith@example.com/jsmith-namespace/devops-secrets.env

{
"PING_IDENTITY_ACCEPT_EULA": "YES",
"PING_IDENTITY_DEVOPS_KEY": "d254....-....-...-...-............",
"PING_IDENTITY_DEVOPS_USER": "jsmith@example.com"
}

SECRET: secrets/jsmith@example.com/jsmith-namespace/certs

{
"tls.crt": "LS0tLS1CRUdJ...a9dk",
"tls.key": "LS0tLS1CRUdJ...38sj"
}
```

Example: Vault secrets .yaml

```
pingfederate-admin:
  vault:
    hashicorp:
      secrets:
        devops-secret.env:
          to-json:
            file: devops-secret.env
        licenses:
          pingaccess-6.2:
            file: pingaccess.lic
            path: /opt/in/some/location/licenses
        test-certs:
          to-json:
            file: test-certs
```

Places the following files into the container.

Example: Container files

FILE: /run/secrets/devops-secret.env

```
PING_IDENTITY_ACCEPT_EULA="YES"
PING_IDENTITY_DEVOPS_KEY="d254....-....-...-...-............"
PING_IDENTITY_DEVOPS_USER="jsmith@example.com"
```

FILE: /opt/in/some/location/licenses/pingaccess.lic

```
Product=PingAccess
Version=6.2
...
```

FILE: /run/secrets/tls.crt

```
LS0tLS1CRUdJ...a9dk
```

FILE: /run/secrets/tls.key

```
LS0tLS1CRUdJ...38sj
```

Using Vault Secrets to Mount base64-Encoded Keystore Files

To pull keystore files from a Vault cluster, create a secret with separate keys for each individual keystore file. The value of each key should be the base64 representation of the file that needs to be mounted. The names of the keys as well as the name of the secret will be used in the Helm values yaml when mounting the keystore files.

Environment variables or other configuration for the product being deployed will need to be set to point to the location where the keystores are being mounted. For example, for PingDirectory:

```
KEYSTORE_FILE=/run/secrets/mykeystore.jks
KEYSTORE_PIN_FILE=/run/secrets/mykeystore.pin
```

Ensure the Vault cluster is accessible to the Kubernetes cluster where your Helm release is being deployed. You can then use the Vault annotations to mount the keystore files in the desired location. For an example, see the "Vault Keystores" example on the Helm examples page.

VolumeMounts Configuration

Provides support for mounting secret or configMap volumes on a workload container.

Global/Product Section

Adds ability to use **secret** and **configMap** data in a container via a VolumeMount. A common use for this configuration is bringing product licenses or scripts into the container.

Example of creating two volume mounts in container from secret and configMap:

```
pingfederate-admin:
   enabled: true
   volumes:
     - name: pf-props
       configMap:
         name: pingfederate-props
     - name: pf-license
       secret:
         secretName: pingfederate-license
   volumeMounts:
     - mountPath: /opt/in/etc/pingfederate.properties
       name: pf-props
     - mountPath: /opt/in/instance/server/default/conf/pingfederate.lic
       name: pf-license
```

> **Note**
>
> Secrets and ConfigMaps must be created in the cluster prior to deploying the helm chart.

In this case, a secret (called `pingfederate-license`) and configMap (called `pingfederate-props`) will bring in a couple of key values (license, hello) and (pf-props) into the container as specific files. The resulting object will look like this:

Example of kubectl describe of pingfederate-admin container

```
Containers:
  pingfederate-admin:
    Mounts:
      /opt/in/etc/pingfederate.properties from pingfederate-props (ro,path="pingfederate.properties")
      /opt/in/instance/server/default/conf/pingfederate.lic from pingfederate-license (ro,path="pingfederate.lic")
Volumes:
  pingfederate-license:
    Type:        Secret (a volume populated by a Secret)
    SecretName:  pingfederate-license
    Optional:    false
  pingfederate-props:
    Type:      ConfigMap (a volume populated by a ConfigMap)
    Name:      pingfederate-props
    Optional:  false
```

Workload Configuration

Kubernetes Workload resources are created depending on configuration values:

- Deployments
- StatefulSets

Global Section

Default yaml is defined in the global `workload` section. Individual products override these defaults based on the required workload.

```
global:
  workload:
    type: Deployment

    deployment:
      strategy:
        type: RollingUpdate
        rollingUpdate:
          maxSurge: 1
          maxUnavailable: 0

    statefulSet:
      partition: 0

      persistentvolume:
        enabled: true
        volumes:
          out-dir:
            mountPath: /opt/out
            persistentVolumeClaim:
              accessModes:
              - ReadWriteOnce
              storageClassName:
              resources:
                requests:
                  storage: 4Gi

    securityContext:
      fsGroup: 9999
    securityContext: {}
```

Workload Parameters	Description
type	One of Deployment or StatefulSet
deployment.strategy.type	One of RollingUpdate or ReCreate
deployment.strategy.rollingUpdate	If type=RollingUpdate
statefulSet.partition	Used for canary testing if n>0
statefulSet.persistentVolume	Provides details around creation of PVC/Volumes (see below)
securityContext	Provides security context details for starting container as different user/group (see below)
securityContext.fsGroup	Sets the group id on fileSystem writes. This is needed especially for mounted volumes (pvs)

Persistent Volumes

For every volume defined in the volumes list, three items will be created in the StatefulSet:

- container.volumeMounts — name and mountPath
- template.spec.volume — name and persistentVolumeClaim.claimName
- spec.volumeClaimTemplates — persistentVolumeClaim

For further details, see the Kubernetes documentation.

Security Context

To run the containers with a different user/group/fsgroup, use the following example to set those details on the deployment/statefulset:

```
global:
  workload:
    container:
      securityContext:
        runAsGroup: 9999
        runAsUser: 9031
        fsGroup: 9999
```

WaitFor

For each product, you can provide a `waitFor` structure indicating the name, service, and timeout (in seconds) for which the container should wait before continuing (default: 300). This setting will inject an initContainer using the PingToolkit wait-for utility that relies on `nc host:port` before continuing.

Example: PingFederate Admin waiting on pingdirectory ldaps service to be available

```
pingfederate-admin:
  container:
    waitFor:
      pingdirectory:
        service: ldaps
        timeoutSeconds: 600
      pingauthorize:
        service: https
        timeoutSeconds: 300
```

- By default, the pingfederate-engine will waitFor pingfederate-admin before it starts.
- By default, the pingaccess-engine will waitFor pingaccess-admin before it starts.

List of Supported Values

These are the values supported in the ping-devops chart. In general, values specified in the global section can be overridden for individual products. The product sections have many global fields overridden by default (workloads, services, etc.).

Global Values

Name	Description	Default
`global.annotations`	Annotations listed, will be added to all Kubernetes resources.	`{}`
`global.labels`	Labels listed, will be added to all Kubernetes resources.	`{}`
`global.envs`	Environment variables listed will be added to the global-env-vars configmap	`{}`
`global.addReleaseNameToResource`	Provides global ability to add names to kubernetes resources. One of `{none, append, prepend}`	`prepend`
`global.ingress.enabled`		`false`
`global.ingress.addReleaseToHost`	Add release to host. One of `{prepend, append, subdomain, none}`	`subdomain`
`global.ingress.defaultDomain`	Replaces with "*defaultDomain*" in host fields	`example.com`
`global.ingress.defaultTlsSecret`	Replaces with "*defaultTlsSecret*" in tls.secretName	
`global.ingress.annotations`		`{}`
`global.ingress.spec.ingressClassName`		
`global.privateCert.generate`	If true, then an internal certificate secret will be created along with mount of the certificate in /run/secrets/internal-cert (creates a tls.crt and tls.key). By default the Issuer of the cert will be the service name created by the Helm Chart. Additionally, the ingress hosts, if enabled, will be added to the list of X509v3 Subject Alternative Name	`false`
`global.privateCert.format`	The format of the certificate to be generated. Used "pingaccess-fips-pem" to generate a valid certificate for running PingAccess in FIPS mode. Any other value will generate a PKCS12 keystore with the generated certificate.	`PKCS12`
`global.privateCert.additionalHosts`	Additional hosts for the cert	`[]`
`global.privateCert.additionalIPs`	Additional IP addresses for the cert	`[]`
`global.masterPassword`	Uses Helm function derivePassword, which uses the master password specification: https://masterpassword.app/masterpassword-algorithm.pdf	
`global.masterPassword.enabled`	Enable master password	`false`

Name	Description	Default
`global.masterPassword.strength`	Master password template. One of `{long, maximum}`	
`global.masterPassword.name`	Defaults to release name	
`global.masterPassword.site`	Defaults to chart name	
`global.masterPassword.secret`	Defaults to release namespace	
`global.vault`	Hashicorp Vault configuration	
`global.vault.enabled`	Enable Vault	`false`
`global.vault.hashicorp.annotations`	Annotation names, which will be appended to 'vault.hashicorp.com/' in the annotation. The vault.hashicorp.annotations.serviceAccountName value will be overwritten by the service account generated for the workload if there is one.	
`global.vault.secretPrefix`	Prefix that will be prepended to any secrets being injected.	`""`
`global.vault.secrets`	Vault secrets to pull in	`{}`
`global.imagePullSecrets`	Repository authentication using secret defined as a docker-registry secret in Kubernetes.	`[]`
`global.image.repository`	Default image registry is not the fully-qualified name of the image Example: image.repository: pingidentity, docker.io, 123.dkr.ecr.us-west-1.amazonaws.com	`pingidentity`
`global.image.repositoryFqn`	Docker image repository fully-qualified name. Overrides image.repository and image.name on the pod image spec Example: image.repositoryFqn: pingidentity/pingfederate, docker.io/my-pingfederate	
`global.image.name`	Default image name MUST be set in child chart Example: image.name: pingfederate	
`global.image.tag`	Default image tag	`2505`
`global.image.pullPolicy`	Default image pull policy	`IfNotPresent`
`global.rbac.generateServiceAccount`	Set to true to generate a service account for the workload.	`false`

Name	Description	Default
`global.rbac.serviceAccountName`	Name of the service account that will be generated. The default value of "*defaultServiceAccountName*" will result in a service account named based on the Helm installation and the specific workload being deployed. If generateServiceAccount and generateGlobalServiceAccount are false, this value can also refer to a service account created outside of Helm.	`defaultServiceAccountName`
`global.rbac.generateRoleAndRoleBinding`	Set to true to generate a Role and RoleBinding corresponding to the workload service account.	`false`
`global.rbac.generateGlobalServiceAccount`	Set to true to generate a service account for the entire installation. This global service account will be used for workloads that do not generate their own service account.	`false`
`global.rbac.generateGlobalRoleAndRoleBinding`	Set to true to generate a Role and RoleBinding corresponding to the global service account for the entire installation.	`false`
`global.rbac.applyServiceAccountToWorkload`	Set to true (the default) to apply to service account to the workload.	`true`
`global.rbac.role`	This yaml will be directly inserted into the generated Role when generateRoleAndRoleBinding and/or generateGlobalRoleAndRoleBinding are true. The rules for the Role can be set here.	`get, watch, and list verbs for the pods resource`
`global.rbac.serviceAccountAnnotations`	Any custom annotations to add to the service account.	
`global.rbac.roleAnnotations`	Any custom annotations to add to the role.	
`global.rbac.roleBindingAnnotations`	Any custom annotations to add to the role binding.	
`global.rbac.serviceAccountLabels`	Any custom labels to add to the service account.	
`global.rbac.roleLabels`	Any custom labels to add to the role.	
`global.rbac.roleBindingLabels`	Any custom labels to add to the role binding.	

Name	Description	Default
`global.externalImage`	Provides ability to use external images for various purposes such as using curl, waitfor, etc. A pingtoolkit image is included by default for running waitFor and generating private cert initContainers. A pingaccess image is also included by default to allow generating an encrypted PEM-formatted cert that is compatible with FIPS mode. Any values specified on the image will be copied directly to the k8s spec for the container, except for the externalImage.{name}.image section, which follows the format of the global.image section. If no image section is specified (the default), the corresponding value from the product values section will be used. For example, if externalImage.pingtoolkit.image is empty, the values from the top-level pingtoolkit.image section will be used.	`{pingtoolkit, pingaccess}`
`global.services`	Services mapping a port to a targetPort on the corresponding container	`{}`
`global.services.clusterExternalDNSHostname`	Value for the external-dns.alpha.kubernetes.io/hostname annotation for the cluster service.	
`global.services.clusterServiceName`	If set, then this name will be used as the cluster service name (i.e clusterService == true).	
`global.services.useLoadBalancerForDataService`	If true, the data service will be created with type: LoadBalancer.	`false`
`global.services.serviceName.dataService`	If true, a ClusterIP service is created reachable within the cluster. A single IP is provided and the service will round-robin across the backend containers	
`global.services.serviceName.clusterService`	If true, a headless service is created, explicitly specifying "None" for the clusterIP. DNS requests to this service will provide one of the IPs of the backend containers	
`global.services.serviceName.containerPort`	Port on the kubernetes container	
`global.services.serviceName.servicePort`	Port available from the kubernetes service. If clusterService=true this port on the cluster service is not really used, as the headless service always maps through to the container port	
`global.services.serviceName.ingressPort`	Port available from the kubernetes ingress	
`global.services.annotations`	Any custom annotations to add to the service.	

Name	Description	Default
`global.services.clusterServiceAnnotations`	Any custom annotations to add to the ClusterIP service.	
`global.services.labels`	Any custom labels to add to the service.	
`global.services.clusterServiceLabels`	Any custom labels to add to the ClusterIP service.	

Workload Values – Deployment and StatefulSet

Name	Description	Default
`global.workload`	Can be Deployment or StatefulSet	`Deployment`
`global.workload.annotations`	Annotations to apply to the template in the workload. To apply top-level annotations to the Deployment or StatefulSet itself, use global.workload.deployment.annotations or global.workload.statefulSet.annotations.	
`global.workload.labels`	Labels to apply to the template in the workload. To apply top-level labels to the Deployment or StatefulSet itself, use global.workload.deployment.labels or global.workload.statefulSet.labels.	
`global.workload.schedulerName`	K8s scheduler	`default-scheduler`
`global.workload.shareProcessNamespace`	Set shareProcessNamespace in the pod spec	`false`
`global.workload.enableServiceLinks`	indicates whether info about services can be added as env variables	`true`
`global.workload.topologySpreadConstraints`	Configuration of pod spread across cluster zones	`[]`
`global.workload.deployment`	Deployment workload configuration	
`global.workload.deployment.strategy`	Deployment pod replacement strategy	
`global.workload.deployment.strategy.type`	Strategy type	`RollingUpdate`
`global.workload.deployment.strategy.rollingUpdate.maxSurge`	Max surge, only applicable for RollingUpdate type	`1`

Name	Description	Default
`global.workload.deployment.strategy.rollingUpdate.maxUnavailable`	Max unavailable, only applicable for RollingUpdate type	`0`
`global.workload.deployment.annotations`	Annotations to apply to the top-level Deployment. To apply annotations to the template within the Deployment, use global.workload.annotations.	
`global.workload.deployment.labels`	Labels to apply to the top-level Deployment. To apply labels to the template within the Deployment, use global.workload.labels.	
`global.workload.statefulSet`	StatefulSet workload configuration	
`global.workload.statefulSet.partition`	Used for canary testing if n>0	`0`
`global.workload.statefulSet.persistentvolume.enabled`	Enable persistent volumes	`true`
`global.workload.statefulSet.persistentvolume.volumes`	For every volume defined in the volumes list, 3 items will be created in the StatefulSet: 1. container.volumeMounts - name and mountPath. 2. template.spec.volume - name and persistentVolumeClaim.claimName. 3. spec.volumeClaimTemplates - persistentVolumeClaim.	`{out-dir}`
`global.workload.statefulSet.persistentvolume.volumes.volumeName.mountPath`	Mount path for the volume	
`global.workload.statefulSet.persistentvolume.volumes.volumeName.persistentVolumeClaim`	volumeClaimTemplate	
`global.workload.statefulSet.podManagementPolicy`	Controls how pods are created during initial scale up, when replacing pods on nodes, or when scaling down. The default behavior is OrderedReady. The Parallel podManagementPolicy allows for starting up and scaling down multiple Pods simultaneously. Updates are not affected. The only products that support Parallel are PingDirectory and PingDataSync, on versions 2209 and later. When using the Parallel policy, consider setting the RETRY_TIMEOUT_SECONDS environment variable to a higher value (it defaults to 180) for the Pods. If the value is too low with many servers starting at once, it may lead to some Pods restarting unnecessarily during the initial workload startup.	`OrderedReady`

Name	Description	Default
`global.workload.statefulSet.annotations`	Annotations to apply to the top-level StatefulSet. To apply annotations to the template within the StatefulSet, use global.workload.annotations.	
`global.workload.statefulSet.labels`	Labels to apply to the top-level StatefulSet. To apply labels to the template within the StatefulSet, use global.workload.labels.	
`global.workload.securityContext`	securityContext for the workload Pod spec. The securityContext defined will be inserted directly into the Pod spec. The user (9031) and group (0) represent the current user and group used with PingIdentity images (except PingDelegator). The fsGroup is required for any workloads that volumeMount a pvc (i.e. StatefulSets). Set as securityContext: null when no generated securityContext is desired.	`fsGroup 0, runAsUser 9031, runAsGroup 0`
`global.clustering.autoscaling`	Configure Horizontal Pod Autoscaling	
`global.clustering.autoscaling.enabled`	Enable Horizontal Pod Autoscaling. If enabled, ensure that proper container.resources values are set and coordinated with the targetCPUUtilizationPercentage or targetMemoryUtilizationPercentage	`false`
`global.clustering.autoscaling.minReplicas`	Autoscaler minimum replicas	`1`
`global.clustering.autoscaling.maxReplicas`	Autoscaler maximum replicas	`4`
`global.clustering.autoscaling.targetCPUUtilizationPercentage`	Target CPU utilization	`75`
`global.clustering.autoscaling.targetMemoryUtilizationPercentage`	Target memory utilization	
`global.clustering.autoscaling.annotations`	Custom annotations for the HPA.	
`global.clustering.autoscaling.labels`	Custom labels for the HPA.	
`global.clustering.autoscaling.behavior`	Custom HPA behavior yaml	`{}`
`global.clustering.autoscalingMetricsTemplate`	Custom HPA metrics yaml	`[]`

Name	Description	Default
`global.container`	Configure the container in the workload Pod spec	
`global.workload.container.securityContext`	securityContext at the container level for the workload. The securityContext defined will be inserted directly into the spec for the main container of the Pod. Container-level securityContext values will overwrite any corresponding values from the Pod-level securityContext.	`allowPrivilegeEscalation: false, capabilities: drop: ALL`
`global.container.replicaCount`	Number of replicas for workload	`1`
`global.container.resources`	container resources yaml to insert into Pod spec	
`global.container.nodeSelector`	nodeSelector yaml to insert into Pod spec	`{}`
`global.container.tolerations`	tolerations yaml to insert into Pod spec	`[]`
`global.container.affinity`	affinity yaml to insert into Pod spec	`{}`
`global.container.terminationGracePeriodSeconds`	termination grace period	`30`
`global.container.envFrom`	envFrom yaml to insert into Pod spec	`[]`
`global.container.env`	Additional environment variables to insert into the Pod spec. Unlike the global.envs values, these will be set directly on the Pod. global.envs values are set in ConfigMaps rather than on the Pod directly. This value allows for setting the valueFrom field for an environment variable when necessary.	`[]`
`global.container.lifecycle`	lifecycle yaml to insert into Pod spec	
`global.container.probes`	probes yaml to insert into Pod spec	`liveness, readiness, and startup probes defined`

Other Global Defaults

Name	Description	Default
`global.license.secret.devOps`	Identify the k8s secret containing the DevOps USER/KEY if used during deployment. pingctl can be used to generate the devops-secret	`devops-secret`

Name	Description	Default
`global.utilitySidecar`	Deploy a utility sidecar for running command-line tools. This sidecar is useful for command line utilities like collect-support-data. The sidecar will remain running alongside the workload, even when the sidecar isn't being used. It does not need to be listed in the includeSidecars value.	
`global.utilitySidecar.enabled`	Enable the utility sidecar	`false`
`global.utilitySidecar.resources`	Set k8s resources yaml for the sidecar spec	`1 CPU and 2g memory limit, 0 CPU and 128Mi memory request`
`global.utilitySidecar.env`	Environment variables for the sidecar	
`global.utilitySidecar.securityContext`	securityContext at the container level for the sidecar. The securityContext defined will be inserted directly into the spec for the sidecar. By default no container securityContext is defined. In Kubernetes when a container-level securityContext is set, it will overwrite any corresponding values from the Pod-level securityContext.	`allowPrivilegeEscalation: false, capabilities: drop: ALL`
`global.includeSidecars`	names of sidecars to include, from the top-level `sidecars` value	`[]`
`global.includeInitContainers`	names of sidecars to include, from the top-level `initContainers` value	`[]`
`global.includeVolumes`	names of sidecars to include, from the top-level `volumes` value	`[]`

Shared Utilities

Name	Description	Default
`sidecars`	Sidecar yaml definitions available to product workload spec	`{}`
`initContainers`	initContainer yaml definitions available to product workload spec	`{}`
`volumes`	volume yaml definitions available to product workload spec for sidecars, initContainers, or main product containers	`{}`
`configMaps`	configMap yaml definitions available to product workload spec for sidecars or main product containers	`{}`

Image/Product Values

Name	Description	Default
`ldap-sdk-tools`	LDAP SDK tools values	
`ldap-sdk-tools.enabled`	Enable LDAP SDK tools deployment	`false`
`pingfederate-admin`	PingFederate admin values	
`pingfederate-admin.enabled`	Enable PingFederate admin deployment	`false`
`pingfederate-admin.cronjob`	CronJobs run a kubectl exec command to run commands on a utility sidecar container. They will also create the necessary ServiceAccount, Role, and RoleBinding to run the jobs	
`pingfederate-admin.cronjob.enabled`	Enable the PingFederate Admin CronJob	`false`
`pingfederate-admin.cronjob.spec`	yaml to insert into the created CronJob spec. If specified, this will override any other specified values for the CronJob.	
`pingfederate-admin.cronjob.spec.jobTemplate`	yaml to override default jobTemplate. If a jobTemplate is not overridden, a default template will be inserted.	
`pingfederate-admin.cronjob.image`	Image to run the Jobs. The image must include kubectl	`bitname/kubectl:latest`
`pingfederate-admin.cronjob.args`	Job arguments	`[]`
`pingfederate-admin.cronjob.podSecurityContext`	securityContext for the pod in the jobTemplate. This will be used if a jobTemplate is not specified.	`null`
`pingfederate-admin.cronjob.podSecurityContext`	securityContext for the container in the jobTemplate. This will be used if a jobTemplate is not specified.	`allowPrivilegeEscalation: false, capabilities: drop: ALL`
`pingfederate-engine`	PingFederate engine values	
`pingfederate-engine.enabled`	Enable PingFederate engine deployment	`false`
`pingfederate-engine.cronjob`	CronJobs run a kubectl exec command to run commands on a utility sidecar container. They will also create the necessary ServiceAccount, Role, and RoleBinding to run the jobs	
`pingfederate-engine.cronjob.enabled`	Enable the PingFederate engine CronJob	`false`

Name	Description	Default
`pingfederate-engine.cronjob.spec`	yaml to insert into the created CronJob spec. If specified, this will override any other specified values for the CronJob.	
`pingfederate-engine.cronjob.spec.jobTemplate`	yaml to override default jobTemplate. If a jobTemplate is not overridden, a default template will be inserted.	
`pingfederate-engine.cronjob.image`	Image to run the Jobs. The image must include kubectl	`bitname/kubectl:latest`
`pingfederate-engine.cronjob.args`	Job arguments	`[]`
`pingfederate-engine.cronjob.podSecurityContext`	securityContext for the pod in the jobTemplate. This will be used if a jobTemplate is not specified.	`null`
`pingfederate-engine.cronjob.podSecurityContext`	securityContext for the container in the jobTemplate. This will be used if a jobTemplate is not specified.	`allowPrivilegeEscalation: false, capabilities: drop: ALL`
`pingdirectory`	PingDirectory values	
`pingdirectory.enabled`	Enable PingDirectory deployment	`false`
`pingdirectory.cronjob`	CronJobs run a kubectl exec command to run commands on a utility sidecar container. They will also create the necessary ServiceAccount, Role, and RoleBinding to run the jobs	
`pingdirectory.cronjob.enabled`	Enable the PingDirectory CronJob	`false`
`pingdirectory.cronjob.spec`	yaml to insert into the created CronJob spec. If specified, this will override any other specified values for the CronJob.	
`pingdirectory.cronjob.spec.jobTemplate`	yaml to override default jobTemplate. If a jobTemplate is not overridden, a default template will be inserted.	
`pingdirectory.cronjob.image`	Image to run the Jobs. The image must include kubectl	`bitname/kubectl:latest`
`pingdirectory.cronjob.args`	Job arguments	`[]`
`pingdirectory.cronjob.podSecurityContext`	securityContext for the pod in the jobTemplate. This will be used if a jobTemplate is not specified.	`null`

Name	Description	Default
`pingdirectory.cronjob.podSecurityContext`	securityContext for the container in the jobTemplate. This will be used if a jobTemplate is not specified.	`allowPrivilegeEscalation: false, capabilities: drop: ALL`
`pingdirectory.services.serviceName.loadBalancerService`	If true, the per-Pod LoadBalancer services enabled with pingdirectory.services.loadBalancerServicePerPod will include this port.	`false`
`pingdirectory.services.loadBalancerServicePerPod`	Set to true to create a separate LoadBalancer service for each individual Pod in the PingDirectory StatefulSet.	`false`
`pingdirectory.services.loadBalancerExternalDNSHostnameSuffix`	Value used for the external-dns.alpha.kubernetes.io/ hostname annotation for the LoadBalancer services. This value will be used as a suffix for the hostname for each individual pod when pingdirectory.services.loadBalancerServicePerPod is set to true.	
`pingdirectoryproxy`	PingDirectoryProxy values	
`pingdirectoryproxy.enabled`	Enable PingDirectoryProxy deployment	`false`
`pingdelegator`	PingDelegator values	
`pingdelegator.enabled`	Enable PingDelegator deployment	`false`
`pingdatasync`	PingDataSync values	
`pingdatasync.enabled`	Enable PingDataSync deployment	`false`
`pingauthorize`	PingAuthorize values	
`pingauthorize.enabled`	Enable PingAuthorize deployment	`false`
`pingauthorizepap`	PingAuthorizePAP values	
`pingauthorizepap.enabled`	Enable PingAuthorizePAP deployment	`false`
`pingaccess-admin`	PingAccess admin values	
`pingaccess-admin.enabled`	Enable PingAccess admin deployment	`false`
`pingaccess-engine`	PingAccess engine values	
`pingaccess-engine.enabled`	Enable PingAccess engine deployment	`false`

Name	Description	Default
`pingcentral`	PingCentral values	
`pingcentral.enabled`	Enable PingCentral deployment	`false`
`pingdataconsole`	PingDataConsole values	
`pingdataconsole.enabled`	Enable PingDataConsole deployment	`false`
`pingdataconsole.defaultLogin`	Default login details for the console	
`pingdataconsole.defaultLogin.server.host`	Default hostname	`pingdirectory-cluster`
`pingdataconsole.defaultLogin.server.port`	Default port	`636`
`pingdataconsole.defaultLogin.username`	Default username	`administrator`
`PingIntelligence`	values	
`pingintelligence.enabled`	Enable PingIntelligence deployment	`false`
`pd-replication-timing`	PingDirectory replication timing values	
`pd-replication-timing.enabled`	Enable PingDirectory replication timing deployment	`false`
`pingtoolkit`	PingToolkit values	
`pingtoolkit.enabled`	Enable PingToolkit deployment	`false`
`testFramework.rbac.serviceAccountImagePullSecrets`	Repository authentication using secrets defined as a docker-registry secrets in Kubernetes.	`[]`

OpenShift Configuration

Openshift is designed to use a randomly generated user ID and group ID (UID/GID) for the `runAsUser` and `fsGroup` fields of the pod- and container-level security contexts.

By default, the security contexts in the chart use values corresponding to the user and group IDs under which the product runs. You can unset the `fsGroup` and `runAsUser` securityContext fields in your custom values, allowing OpenShift to set them as expected.

Unset `fsGroup` and `runAsUser` at the Pod Level

In the global section of the values.yaml file, add the following stanza:

```
global:
  workload:
    securityContext:
      fsGroup: null
      runAsUser: null
```

This will unset `fsGroup` and `runAsUser` in the pod-level security context. Pods that require initContainers will have to also unset `runAsUser` in the container-level security context.

initContainers: Unset `runAsUser` at the Container Level

Some of the product deployments use initContainers for various operations, such as waiting for other services to be available or configuration actions. These containers, while part of the workload, have the security context set at the container level, not the pod level. The values listed above apply only to the pod-level security context. To unset `runAsUser` for any pingtoolkit initContainers so Openshift can take over, also add the following stanza:

```
global:
  externalImage:
    pingtoolkit:
      securityContext:
        runAsUser: null
```

For example, here is a complete block for configuring pingaccess-admin with a `waitFor` initContainer:

```
global:
  workload:
    securityContext:
      fsGroup: null
      runAsUser: null
  externalImage:
    pingtoolkit:
      securityContext:
        runAsUser: null

  pingaccess-admin:
    enabled: true
    privateCert:
    generate: true
    envs:
      SERVER_PROFILE_URL: https://github.com/pingidentity/pingidentity-server-profiles.git
      SERVER_PROFILE_PATH: baseline/pingaccess
    container:
      waitFor:
        pingfederate-engine:
          service: https
          timeoutSeconds: 300
```

Examples

Helm Chart Examples

The example deployments of various configurations and combinations of Ping products with these charts are provided on the Ping Identity DevOps Portal.

> **Note**
>
> Install and configure Helm according to the Getting Started page before trying the examples.

How To

Updating Product Image Tags

This page describes recommendations for updating products and image tags in a Helm installation. The focus is on what is needed to move from one Docker image tag to another in the Helm chart, with some specific steps to follow for certain products. This page does not cover specifics on how to test upgrades or details on how to employ a blue-green update strategy. These processes will depend on the environment where the chart is deployed.

> **Warning**
>
> Always test upgrades in a non-production environment first.

Before updating tags, make sure that you have exported any configuration from your servers and stored it in a server profile. This ensures no configuration is lost when pods are restarted. Information on how to save configuration into a server profile can be found in the How To section of the DevOps documentation.

The upgrade process depends on the specific product within the Helm chart that is being upgraded.

Blue-Green Update

A simple way to handle version updates is via a blue-green deployment. In this strategy, a full second deployment of the workloads being updated is deployed on the new version, and then traffic is switched from the original deployment to the new one. When following this strategy, be sure that any necessary configuration is captured in a server profile, so that it is maintained in the fresh deployment.

This strategy is particularly suited for stateless applications. Stateful applications like PingDirectory and PingDataSync would require further steps to ensure any data in the current deployment is maintained in the newer deployment.

In-Place Update

Tags and product versions can also be updated in place. The required process depends on the product being updated.

Simple Tag Update

For some products, simply updating the tag is all that is needed. For example, updating from PingDirectory `8.3.0.5-latest` to PingDirectory `9.0.0.0-latest` can be done by updating the tag in values.yaml.

```
pingdirectory:
  enabled: true
  image:
    tag: 8.3.0.5-latest
```

becomes

```
pingdirectory:
  enabled: true
  image:
    tag: 9.0.0.0-latest
```

The active Helm release can be updated after setting the new image tag.

```
helm upgrade --install <releasename> pingidentity/ping-devops -f <updated values.yaml>
```

Update logic will be automatically handled by hook scripts on container startup. This is true for PingDirectory, PingDirectoryProxy, PingDataSync, and PingAuthorize.

For products that are not run as a StatefulSet, the tags can generally be updated without any additional update steps, after the server profile is updated and saved. If the server profile is not up to date with the configuration on the running pod(s), then the configuration could be lost on a tag update.

Product Updates Requiring Manual Steps

Some products have more specific upgrade processes that need to be followed when the product version is being updated, such as updating from PingFederate 10.3.7 to PingFederate 11.0.3.

PingFederate

See the DevOps documentation for instructions on upgrading PingFederate.

By default, `pingfederate-admin` and `pingfederate-engine` each run as Deployments in the chart. If you are running a maintenance upgrade (from PingFederate version x.x.1 to x.x.2 for example), it may be sufficient to export the configuration from the previous version and pull that same configuration via server profile when starting up the new version.

> **Warning**
>
> When upgrading a major or minor version, the upgrade steps at the above link should be followed.

PingAccess

PingAccess follows a similar process to the above instructions for PingFederate.

By default, `pingaccess-admin` runs as a StatefulSet in the chart, so the upgrade utility must be run to update the files in the persistent volume. Once the admin is updated, the engines (which run as Deployments by default) can be restarted with the new tag. See the PingAccess documentation for more information.

PingAuthorizePAP

Ensure your policies are backed up along with your server profile before updating the tag.

Release Notes

Current Release

Release 0.11.8 (June 4, 2025)

Features

- Updated default global image tag to `2505`.
- Support creation of CronJob resources for PingFederate, similar to the existing support available for PingDirectory (PDI-2218).

Enhancements

- Allow setting securityContext for CronJob and utility sidecar (PDI-2208).

Bug fixes

- Fix incorrect indentation of global configMap name when setting global annotations (PDI-2215).

Previous Releases

Release 0.11.7 (May 1, 2025)

Features

- Updated default global image tag to `2504`.

Release 0.11.6 (April 2, 2025)

Features

- Updated default global image tag to `2503`.

Release 0.11.5 (March 3, 2025)

Features

- Updated default global image tag to `2502`.

Release 0.11.4 (February 14, 2025)

Features

- Updated default global image tag to `2501.1`.

Release 0.11.3 (February 3, 2025)

Features

- Updated default global image tag to `2501`.

Bug fixes

- Prevented unexpected service port entries from being created on workloads when setting service labels or annotations. (PDI-2182)

Release 0.11.2 (January 3, 2025)

Features

- Updated default global image tag to `2412`.

Release 0.11.1 (December 3, 2024)

Features

- Updated default global image tag to `2411`.

Release 0.11.0 (November 20, 2024)

Features

- Updated default global image tag to `2410`.

Enhancements

- Added supported values for specifying more fine-grained annotations and labels for workloads, workload pod templates, services, HPA, and RBAC objects. Annotations can now be specified for these resources individually, rather than relying on global annotations that apply to all resources. Here is an example showing the new values that can be set for annotations (analagous values are available to control labels):

```
global:
  annotations:
    globalAnnotation: val

  workload:
    annotations:
      globalWorkloadAnnotation: val
    statefulSet:
      annotations:
        globalSSAnnotation: val
    deployment:
      annotations:
        globalDepAnnotation: val

  rbac:
    generateGlobalServiceAccount: true
    generateGlobalRoleAndRoleBinding: true
    serviceAccountAnnotations:
      globalServiceAccountAnnotation: val
    roleAnnotations:
      globalRoleAnnotation: val
    roleBindingAnnotations:
      globalRoleBindingAnnotation: val

  services:
    annotations:
      globalServiceAnnotation: val
    clusterServiceAnnotations:
      globalClusterServiceAnnotation: val

pingdirectory:
  enabled: true
  annotations:
    pdAnnotation: val
  workload:
    annotations:
      pdWorkloadAnnotation: val
    statefulSet:
      annotations:
      pdSSAnnotation: val
  rbac:
    generateServiceAccount: true
    generateRoleAndRoleBinding: true
    role:
      rules:
      - apiGroups: [""]
        resources: ["secrets"]
        verbs: ["get", "list"]
    serviceAccountAnnotations:
      pdServiceAccountAnnotation: val
      roleAnnotations:
        pdRoleAnnotation: val
      roleBindingAnnotations:
        pdRoleBindingAnnotation: val
    services:
      annotations:
        pdServiceAnnotation: val
      clusterServiceAnnotations:
        pdClusterServiceAnnotation: val
```

Bug fixes

- Fixed global annotations not applying for RBAC objects.

Release 0.10.9 (October 1, 2024)

Features

- Updated default global image tag to `2409`.

Release 0.10.8 (September 4, 2024)

Features

- Updated default global image tag to `2408`.

Release 0.10.7 (August 6, 2024)

Features

- Updated default global image tag to `2407`.

Release 0.10.6 (July 2, 2024)

Features

- Updated default global image tag to `2406`.

Release 0.10.5 (June 5, 2024)

Features

- Updated default global image tag to `2405`.

Release 0.10.4 (May 1, 2024)

Features

- Updated default global image tag to `2404`.

Release 0.10.3 (March 29, 2024)

Features

- Updated default global image tag to `2403`.

Release 0.10.2 (March 1, 2024)

Features

- Updated default global image tag to `2402`.

Release 0.10.1 (February 5, 2024)

Bug Fixes

- Fixed templating failure when not specifying `ingress.spec.ingressClassName` with Ingress enabled.

Release 0.10.0 (January 31, 2024)

Features

- Updated default global image tag to `2401`.
- Added support for setting environment variables in utility sidecar pods, with the `utilitySidecar.env` value.
- Added support for setting ingressClassName in Ingress specs with the `ingress.spec.ingressClassName` value.

Enhancements

- Updated generated PVC definitions to include annotations.

Release 0.9.22 (December 29, 2023)

Features

- Updated default global image tag to `2312`.

Release 0.9.21 (December 4, 2023)

Features

- Updated default global image tag to `2311`.

Defects

- Updated the workload template to avoid setting `replicas` when autoscaling is enabled.
- Improved capabilities checks for `apiVersion` field to avoid issues with prerelease Kubernetes versions.

Release 0.9.20 (November 2, 2023)

Features

- Updated default global image tag to `2310`.

- Added environment variables for PingDirectoryProxy to support enabling automatic server discovery.

Defects

- Updated the CronJob template to handle the switch from `batch/v1beta1` to `batch/v1` in Kubernetes 1.25.

Release 0.9.19 (September 6, 2023)

Features

- Updated default global image tag to `2308`.

Release 0.9.18 (August 28, 2023)

Resolved Defects

- Fixed incorrect yaml formatting when setting `testFramework.rbac.serviceAccountImagePullSecrets`.

Release 0.9.17 (August 25, 2023)

Features

- Added support for setting `imagePullSecrets` in workloads.
- Added support for setting `testFramework.rbac.serviceAccountImagePullSecrets` to add secrets to the testFramework service account.

Release 0.9.16 (August 2, 2023)

Features

- Updated default global image tag to `2307`.

Release 0.9.15 (July 13, 2023)

Features

- Updated default global image tag to `2306`.

Enhancements

- Updated template to allow setting a custom workload type when using a HorizontalPodAutoscaler.

Release 0.9.14 (June 2, 2023)

Features

- Updated default global image tag to `2305`.

Release 0.9.13 (May 4, 2023)

Features

- Updated default global image tag to `2304`.

Release 0.9.12 (April 3, 2023)

Features

- Updated default global image tag to `2303`.
- Updated the workload `topologySpreadConstraints` field to automatically set `matchLabels` to match the workload labels.

Release 0.9.11 (March 3, 2023)

Features

- Updated default global image tag to `2302`.
- Added default environment variables to `pingdirectoryproxy` to support joining a PingDirectory topology.

Release 0.9.10 (February 3, 2023)

Features

- Updated default global image tag to `2301`.
- Updated the securityContext defaults for Pods and containers in the ping-devops Helm chart to satisfy the "restricted" Pod Security Standard in Kubernetes.
- Added support for running a separate LoadBalancer service for each PingDirectory pod. This may be useful when running across multiple regions when using VPC peering isn't possible.

Resolved Defects

- Updated the HorizontalPodAutoscaler API to use the correct value for Kubernetes versions greater than 1.23.

Release 0.9.9 (January 3, 2023)

Features

- Updated default global image tag to `2212`.
- Removed `pingdatagovernance` and `pingdatagovernancepap` from the chart. Use `pingauthorize` and `pingauthorizepap` instead.

Release 0.9.8 (December 5, 2022)

Features

- Updated default global image tag to `2211`.
- Custom annotations can now be specified for Services.

Defects

- Fixed HorizontalPodAutoscaler autoscalingMetricsTemplate being inserted in the wrong location in the generated yaml.
- Fixed the documentation in values.yaml referring to `pingdirectory.cronjob.jobspec` rather than the correct value `pingdirectory.cronjob.jobTemplate`.

Release 0.9.7 (November 2, 2022)

Features

- Updated default global image tag to `2210`.

Release 0.9.6 (October 4, 2022)

Features

- Updated default global image tag to `2209`.
- Added support for deploying a HorizontalPodAutoscaler for pingaccess-engine, pingfederate-admin, pingdelegator, pingauthorize, pingauthorizepap, pingcentral, and pingdataconsole. Previously, deploying a HorizontalPodAutoscaler was only supported for pingfederate-engine.
- Added support for setting the podManagementPolicy for StatefulSet workloads. The default policy is OrderedReady. The Parallel policy allows for starting up multiple Pods of the StatefulSet simultaneously, improving initial deployment times. Parallel startup is only supported with PingDirectory and PingDataSync, and only with images version 2209 and newer.

Release 0.9.5 (September 1, 2022)

Features

- Updated default global image tag to `2208`.
- Added support for setting container-level securityContext values for the main container of each workload. By default no container-level securityContext will be set. A container-level securityContext isn't necessary if the values from the Pod-level securityContext are sufficient.
- Added support for setting the topologySpreadConstraints field on workloads.
- Added support for setting the enableServiceLinks field on workloads.

Documentation

- Added an example for mounting keystore secrets with Vault.
- Added an example for mounting secrets with CSI volumes (which can be used for various storage systems including AWS secrets manager)
- Fixed Helm RBAC example using an invalid serviceAccountName for pingauthorize.
- Added a doc page describing how to update product versions.
- Added example docs for deploying PingDirectory and PingFederate in a multi-region environment with Helm.

Resolved Defects

- Removed support for apache-jmeter, since it is better suited to run as a job than as a long-running workload.

Release 0.9.4 (August 5, 2022)

Features

- Updated default global image tag to `2207`.
- Added support for apache-jmeter

Resolved Defects

- Fixed an issue making it impossible to use an existing service account (an account not managed by the Helm chart) for a workload. An existing service account can now be used by specifying the {product}.rbac.serviceAccountName field while leaving {product}.rbac.generateServiceAccount set to the default false value. See the PingAuthorize section of the updated RBAC example.

Release 0.9.3 (July 1, 2022)

Features

- Updated default global image tag to `2206`.
- Added support for PingIntelligence.
- Updated the Helm chart to support generating ServiceAccounts, Roles, and RoleBindings for a workload. These can be generated globally (one common to each workload) or individually for each workload. By default, none will be generated.

  These can be controlled with the global.rbac (or {product}.rbac) section. To generate a common ServiceAccount usable by all workloads, use global`.rbac.generateGlobalServiceAccount`. Use `rbac.generateServiceAccount` to generate separate ServiceAccounts for individual workloads. Similarly, use `global.rbac.generateGlobalRoleAndRoleBinding` and `rbac.generateRoleAndRoleBinding` for creating a Role and RoleBinding.

  Set rbac.applyServiceAccountToWorkload to true to set the account on the Deployment or StatefulSet. The name of the ServiceAccount will be autogenerated unless the rbac.serviceAccountName field is set. The specific Role yaml can be provided in rbac.role.

The Vault default has changed. The Vault serviceAccount will now default to the autogenerated account for the workload, instead of the previous default of "vault-auth". This can be overriden by setting the vault.hashicorp.annotations.serviceAccountName value.

See the table with sample Helm chart value files found on the Ping Identity Devops Portal for an example.

- Added a default empty `global.labels` section.

Release 0.9.2 (June 2, 2022)

Features

- Updated default global image tag to `2205`.
- Added support for providing a null securityContext for a workload, which is useful for OpenShift security context constraints.
- Added support for enabling Ingress for pingdirectoryproxy and pingdatasync.
- Updated pingdirectoryproxy to be a StatefulSet by default in the Helm charts, with the persistent volume disabled. This supports having consistent proxy pod names.
- Added a new `privateCert.format` field, which can be set to “pingaccess-fips-pem” to generate a cert that can be used by PingAccess when running in FIPS mode. The cert is generated by a temporary PingAccess initContainer, as PingAccess requires a specific format for certs when in FIPS mode that must be generated from PingAccess itself. Leaving the field blank or setting it to any other value will generate a cert in the same manner as before: adding the key pair to a PKCS12 keystore file.
- Updated the externalImage values section to expect the same format for `image:` as the individual products (repository, image name, tag, etc. are provided separately). If no image values are specified for an externalImage, the corresponding defaults from the main product section will be used. For example, if `global.externalImage.pingtoolkit.image` is empty, then the values from the top-level `pingtoolkit.image` section will be used.

Release 0.9.1 (May 5, 2022)

Features

- Updated default global image tag to `2204`.
- Updated the PingDataSync env vars ConfigMap to include variables needed to enable failover between servers. Failover will be enabled when deploying two or more PingDataSync replicas
- Reduced utilitySidecar resource requests.

Resolved Defects

- Updated the image.repositoryFqn field to be consistent with the other fields under image. Previously, repositoryFqn was expected at the same level as the image: section, now it is expected within the image: section like other fields (tag, pullPolicy, etc.). The image tag must now be provided separately from the repositoryFqn. The repositoryFqn should only be the name of the repository, not the tag of the specific image.
- Fixed a version check in the Helm chart for choosing the correct k8s API for Ingress. The version check was previously failing on EKS clusters due to the format EKS uses for the cluster version.

Release 0.9.0 (April 1, 2022)

Features

- Default global image tag updated to `2203`.
- Customizability on Cronjob and Utility Sidecar:
  - Override jobTemplate in CronJob now available.
  - Override image used in utilitySidecar now available.
- Updated the default PingDataSync workload in the Ping devops Helm charts to use a StatefulSet rather than a Deployment. This ensures that the sync-state.ldif file is maintained between pod restarts.

Release 0.8.9 (March 17, 2022)

Features

- Edit from 0.8.8 release. Previously, the image fully qualified name also included the image tag, which was then duplicated upon deployment when "tag" value present.

Release 0.8.8 (March 16, 2022)

Features

- Added support for fully qualified image location. For more information go to the image section in our values.yaml.

```
image:
  repository: pingidentity
  repositoryFqn:
  name:
  tag: "2202"
  pullPolicy: IfNotPresent
```

Release 0.8.7 (March 11, 2022)

Features

- Corrected default global image tag updated to `2202`.

Release 0.8.6 (March 3, 2022)

Features

- Default global image tag updated to `2202`.

Release 0.8.5 (February 7, 2022)

Features

- PingCentral now supported. Example values application found here.

Issues Resolved

- Issue #119 — Workload template not honoring false values from values.yaml. Previously, false did not overwrite true in the Ping Identity Helm Chart template. This fix in _merge-util.tpl will resolve multiple cases within the Ping Identity Helm Chart.

```
{{- $globalValues := deepCopy $top.Values.global -}}
{{- $prodValues := deepCopy (index $top.Values $prodName) -}}
{{- $mergedValues := mergeOverwrite $globalValues $prodValues -}}
```

- Issue #264 — Update default global.image.tag to `2201`.

Release 0.8.4 (January 7, 2022)

Issues Resolved

- Fixed an issue that caused installation to fail when enabling `pingtoolkit`.

Release 0.8.3 (January 6, 2022)

Features

- Document supported values.

Issues Resolved

- Issue #233 — Ingress - semverCompare now retrieves correct K8 version for applying the correct apiVersion.

```
{{- if semverCompare ">=1.19.x" $top.Capabilities.KubeVersion.Version }}
```

- Issue #254 — Update default global.image.tag to `2112`.

Release 0.8.2 (December 17, 2021)

- Issue #238 — Added support for running a utility sidecar alongside a product workload

  The `utilitySidecar` field under a given product can be used to run a sidecar container that will permanently alongside the product container. This sidecar can be used for utility command-line processes, such as running the collect-support-data tool or running a backup.

An example can be found in the docs/examples/pingdirectory-backup directory for running a PingDirectory backup every 6 hours via a CronJob.

```
pingdirectory:
  workload:
    shareProcessNamespace: true
  utilitySidecar:
    enabled: true
```

- Issue #247 — Update default global.image.tag to `2111.1`.

Release 0.8.1 (December 6, 2021)

- Issue #240 — Fixed failure on installation of 0.8.0 due to missing PingDirectory HTTP port value.

Release 0.8.0 (December 6, 2021)

- Issue #229 — Support for shareProcessNamespace in pod spec.

  A PingDirectory utility sidecar container needs to share the process namespace with the main PingDirectory container running in the same pod in order to get useful output out of tools like jps. More support to come on the utility sidecar in future Helm release.

- Issue #232 — Update default global.image.tag to 2111

- Issue #239 — Support for custom container arguments

```
pingfederate-admin:
  enabled: true
  container:
    args: ["start-server","tail -f /dev/null"]
```

- Issue #232 — Update default global.image.tag to 2111

- Issue #240 — Allow specifying PingDirectory HTTPS port in values

```
pingdirectory:
  enabled: true
  services:
    https:
      containerPort: 8443
```

Release 0.7.9 (December 1, 2021)

- Issue #223 — Support for HPA Scaling Behavior

```
clustering:
  autoscaling:
    enabled: true
    behavior:
      scaleDown:
        stabilizationWindowSeconds: 300
        policies:
        - type: Percent
          value: 100
          periodSeconds: 15
      scaleUp:
        stabilizationWindowSeconds: 0
        policies:
        - type: Percent
          value: 100
          periodSeconds: 15
        - type: Pods
          value: 4
          periodSeconds: 15
          selectPolicy: Max
```

- Issue #231 — Helm test image pull policy no longer hard-coded in `helm-charts/charts/ping-devops/templates/pinglib/_tests/tpl`.

```
- imagePullPolicy: IfNotPresent
```

- Issue #233 — Cluster service for pingaccess-admin in Multi-region Support for multi-region PingAccess deployment without using an ingress. The headless service is an effective way to share the pod id across clusters.

```
pingaccess-admin:
enabled: true
privateCert:
  generate: true
envs:
  SERVER_PROFILE_URL: https://github.com/pingidentity/pingidentity-server-profiles.git
  SERVER_PROFILE_PATH: baseline/pingaccess
container:
  replicaCount: 1
  waitFor:
    pingfederate-engine:
      service: https
services:
  https:
    servicePort: 9000
    containerPort: 9000
    ingressPort: 443
    dataService: true
    clusterService: true
  clusterconfig:
    servicePort: 9090
    containerPort: 9090
    ingressPort: 443
    dataService: true
  clusterExternalDNSHostname: pingaccess-admin.usa.ping-multi-cluster.com
```

Release 0.7.8 (November 2, 2021)

- Issue #213 — Removed default SERVER_PROFILE variables from values.yaml

```
envs:
- SERVER_PROFILE_URL:
- SERVER_PROFILE_PATH:
```

- Issue #216 — Add option to generate a master password for ping services

  In the interest of better security practice, this enhancement provides the ability to generate this password via the derivedPassword function in helm. With this, several items can be used by default and overridden by the deployer to generate a secure password. When it generates the password:

- A note will be added to the NOTES (see below)
- The password will be set into the global configmap PING_IDENTITY_PASSWORD. (we may want to use a secret instead)

  NOTES (see the generated password as well as the WARNING)

```
$ helm upgrade --install test-pw pingidentity/ping-devops  --set global.masterPassword.enabled=true
    NOTES:
    #-------------------------------------------------------------------------------
# Ping DevOps
#
# Description: Ping Identity helm charts - 09/18/21
#-------------------------------------------------------------------------------
# WARNING: Master Password has been requested and generated.  This is intended to
#          generate a password for DEVELOPMENT PURPOSES ONLY.  This password will be
#          assigned to the PING_IDENTITY_PASSWORD unless overridden by the values.
#
#          PING_IDENTITY_PASSWORD: **************
#-------------------------------------------------------------------------------
```

The values used to drive the creation of this password are:

values.yaml

```
global:
      ############################################################
      # Master Password Generation
      #
      # Uses Helm function derivePassword, which uses the master password
      # specification: https://masterpassword.app/masterpassword-algorithm.pdf
      #
      #      masterPassword.enabled: {true | false}
      #      masterPassword.strength: {master password template: long | maximum}
      #      masterPassword.name: {defaults to .Release.Name}
      #      masterPassword.site: {defaults to .Chart.Name}
      #      masterPassword.secret: {defaults to .Release.Namespace}
      ############################################################
      masterPassword:
        enabled: false
        strength: long
        name:   # default - .Release.Name
        site:   # default - .Chart.Name
        secret: # default - .Release.Namespace
```

+ As shown in the example above, a deployer only needs to provide the global.masterPassword.enabled=true to have it generated.

- Issue #221 — PingDirectory service.x.containerPort updates to LDAPS_PORT environment variable.
- Issue #222 — Update default global.image.tag to `2110`.
- Issue #224 — External Hostname Annotations on PD data service . == Release 0.7.7 (Oct 7, 2021)
- Issue #217 — Update default security context group id to root (0).

```
global:
  workload:
    securityContext:
      fsGroup: 0
      runAsUser: 9031
      runAsGroup: 0
```

- Issue #218 — Update default global.image.tag to `2109`.

Release 0.7.6 (September 18, 2021)

- Issue #209 — Fix incorrect default ldap-sdk-tools probe exec commands.
- Issue #210 — Add helm-chart product/image pingtoolkit.
- Issue #211 — Allow for schedulerName to be provide on workloads (pods).

Release 0.7.5 (August 30, 2021)

- Issue #206 — Bump default image tag to `2108`.

Release 0.7.4 (August 26, 2021)

- Issue #196 — Set initContainer settings from values.yaml instead of hard coded templates.

  This issue was created since the initContainer resources were hard coded in the template, not allowing the implementor to provide their own values, causing issues when trying to deploy the pingfederate-engine in openshift.

  Moving a lot of the hard coded yaml out of the template files into the default values.yaml file. This will give the implementor full control of how the initContainer runs.

  One breaking change with the values.yaml if anyone has overridden, is that the `{image name}` in the `global.externalImage.{name}: {image name}` value is moved into a map. The default pingtoolkit externalImage looks like:

```
global:
  externalImage:
    pingtoolkit:
      image: pingidentity/pingtoolkit:2107
      imagePullPolicy: IfNotPresent
      resources:
        limits:
          cpu: 1m
          memory: 128Mi
        requests:
          cpu: 500m
          memory: 64Mi
      securityContext:
        allowPrivilegeEscalation: false
        capabilities:
          drop:
          - ALL
        readOnlyRootFilesystem: true
        runAsNonRoot: true
        runAsUser: 9031
        runAsGroup: 9999
```

- Issue #203 — testFramework - Support multiple waitFor products in testSteps.

  When there are two waitFor's together, allow for combining them to run them within same initContainer, with a definition like:

  ```
  testSteps:
    - name: 01-wait-for
      waitFor:
        pingfederate-admin:
          service: https
        pingfederate-engine:
          service: https
  ```

  creating a couple of initContainers of:

  ```
  initContainers:
    - name: 01-wait-for-pingfederate-admin
      ...
    - name: 01-wait-for-pingfederate-engine
      ...
  ```

Release 0.7.3 (August 24, 2021)

- Issue #194 — Change default envs for pingauthorize/pingauthorizepap.

  The current envs for pingauthroize in the values.yaml file are:

```
envs:
  SERVER_PROFILE_URL: https://github.com/pingidentity/pingidentity-server-profiles.git
  SERVER_PROFILE_PATH: paz-pap-integration/pingauthorize
  SERVER_PROFILE_PARENT: PAZ
  SERVER_PROFILE_PAZ_URL: https://github.com/pingidentity/pingidentity-server-profiles.git
  SERVER_PROFILE_PAZ_PATH: baseline/pingauthorize
```

Just a side note here, the `baseline/pingauthorize` PATH includes a connection to pingdirectory, which will cause this to fail (pingauthorize https will return a 503).

If someone wants to override these, they need to be sure to uset/override the SERVER_PROFILE_PARENT variable, so the parent profiles aren't brought in.

The better default values.yaml should probably be:

```
envs:
  SERVER_PROFILE_URL: https://github.com/pingidentity/pingidentity-server-profiles.git
  SERVER_PROFILE_PATH: getting-started/pingauthorize
```

For pingauthorizepap, it should have a default SERVER_PROFILE variables as empty, as no SERVER_PROFILE is needed by default.

- Issue #198 — testFramework: Support full definition of initContainers attributes in testSteps and finalStep.

  Update the testFramework to pull in all attributes of the testSteps and finalStep into the init containers and final container. This allow for setting any resource, imagePullPolicy, ...

  This came about as there was no way to set resource or imagePullPolicy details.

  With this change, will be adding a couple of defaults into the value.yamls file for the finalStep:

```
  finalStep:
    name: 99-completion
    image: busybox
    imagePullPolicy: IfNotPresent
    command:
      ...
    resources:
      limits:
        cpu: 500m
        memory: 128Mi
      requests:
        cpu: 1m
        memory: 64Mi
```

Release 0.7.2 (August 13, 2021)

- Issue #191 — Change variable PF_ADMIN_BASEURL to PF_ADMIN_PUBLIC_BASEURL.

  Release 0.7.2 created the new variable `PF_ADMIN_BASEURL`. Due to the current user of the same variable with added `_PUBLIC_`, the actual variable name needs to be `PF_ADMIN_PUBLIC_BASEURL`.

Release 0.7.1 (August 13, 2021)

- Issue #187 — Create the PUBLIC hostname/ports in the global env vars configmap all the time.

  Currently, the PUBLIC hostname/ports in the global env vars configmap are created if and only if the ingress in enabled.

  Normally, this would be fine, except that some of the products (i.e PingFederate) use the PUBLIC environment variable to setup items like BASE URLs and redirects for the browser. This is required for use cases when there is no ingress, but the user creates a port forward, as well as testing with no ingresses.

  So, if no ingress is created, then the PUBLIC_HOSTNAMES should be set to localhost and the PUBLIC_PORT_* should be set to the same port as the contianerPort.

  If ingress is used, then the functionality will not be changed, and the public hostname will be constructed as well as the public ingressPort.

- Issue #188 — Add the PF_ADMIN_BASEURL environment variable to the pingfederate admin/engine configmaps

  With the 10.3 release of PingFederate, there is a variable used to provide redirect links called the PF_ADMIN_BASEURL. This needs to be set by the helm chart, as it will either be a public host or localhost, depending on if the ingress is available. The container has no idea which it should be as it doesn't have insight into the environment it's running.

  If ingress is enabled, an example for this variable is:

  `PF_ADMIN_BAESURL=https://pingfederate-admin.example.com`

  If ingress is not enable, an example for this variable:

  `PF_ADMIN_BASEURL=https://localhost:9999`

Release 0.7.0 (August 9, 2021)

- Issue #184 — Create default ServiceAccount/Role/RoleBinding for testFramework.

  To allow for a role to be created during testing, an rbac section is added to the testFramework allowing for the definition of that Role. If enabled, it will create a ServiceAccount, Role and RoleBinding using the same naming rules of resources and add that serviceAccount to the test pod.

  testFramework default rbac set to:

```
  ##########################################################
  # If rbac is enabled, this will create:
  #   - serviceAccount
  #   - role
  #   - roleBinding (between serviceAccount and role)
  #
  # and apply the serviceAccount to the pod in the tests.
  # The names for these resources will be named using the
  # naming rules for all resources including the ReleaseName
  ##########################################################
  rbac:
    enabled: true
    role:
      rules:
      - apiGroups:
        - '*'
        resources:
        - '*'
        verbs:
        - '*'
```

Release 0.6.9 (August 6, 2021)

- Issue #179 — Bump default image tag to 2107 Issue #182 Set default startupProbe.timeoutSeconds to 5.
- Issue #180 — Enhance testFramework to support additional pod level configurations.

  When using the testFramework there are additional pod level config items that need to be provided (i.e. serviceAccountName) along with the existing securityContext.

  To allow for any item to be configured, we should add a testFramework.pod that will pull in all items into the testFramework pod definition.

  Example:

```
testFramework:
  ##########################################################
  # Pod information to include
  #
  # Examples:
  #   securityContext for all containers
  #   serviceAccount for all containers
  ##########################################################
  pod:
    securityContext:
      runAsUser: 1000
      runAsGroup: 2000
    serviceAccount: serviceaccount-name
```

> **Warning**
>
> This will be a breaking change for anyone who has created a `testFramework.securityContext`. If this is the case, they need to add pod in front of `securityContext`.

Release 0.6.8 (July 29, 2021)

- Issue #175 — Invalid ingress resources on Kubernetes clusters > 1.18.

  During resoluton of issue #170 providing support for ingress apiVersion v1, the necessary ingress yaml fields wearn't updated to relfect that new version. This is a fix. The backend definition of the Ingress will now reflect the proper definition based on a v1 or v1beta1 apiVersion.

  Example: If KubeVersion > 1.18

  ```
  service:
    name: https
    port:
      number: 443
  ```

  Example: If KubeVersion ⇐ 1.18

  ```
  serviceName: https
  servicePort: 443
  ```

  Additionally, adding the pathType for all versions as it is now required in ingress v1.

Release 0.6.7 (July 28, 2021)

- Issue #170 — Update Ingress resource kind.

  If kubernetes version is >1.18, setting the ingress apiVersion to `v1`. Otherwise, current default will be used `v1beta1`.

- Issue #171 — Reevaluate Lifecycle probes.

  Adding startupProble as well as re-organizing how the probes are defined, allowing the deployer to use standard k8s probe definitions out of the box.

  - Moving the probes section under global.container
  - Changing names: (liveness → livenessProbe, readiness → readinessProbe)
  - Adding startupProbe

  The new default looks like:

```
############################################################
# Probes
#
# Probes have a number of fields that you can use to more precisely control the
# behavior of liveness and readiness checks.
#
# https://kubernetes.io/docs/tasks/configure-pod-container/configure-liveness-readiness-startup-probes/
############################################################
probes:
  livenessProbe:
    exec:
      command:
        - /opt/liveness.sh
    initialDelaySeconds: 30
    periodSeconds: 30
    timeoutSeconds: 5
    successThreshold: 1
    failureThreshold: 4
  readinessProbe:
    exec:
      command:
        - /opt/readiness.sh
    initialDelaySeconds: 30
    periodSeconds: 5
    timeoutSeconds: 5
    successThreshold: 1
    failureThreshold: 4
  startupProbe:
    exec:
      command:
        - /opt/liveness.sh
    periodSeconds: 10
    failureThreshold: 90
```

Breaking Changes

This is a breaking change if anyone has overriding probes in their own values file. The fix is simply move their definition of their probes to live under global.container or the (productName).container, as well as adding "Probe" to the definition.

Release 0.6.6 (July 7, 2021)

- Issue #160 — Change default image tag to `2106`.
- Issue #166 — Add securityContexts to testFramework containers.
  - Adding ability to provide a securityContext at the following levels:
  - Changing the default finalStep image to busybox

```
testFramework:
  ...
  ########################################################
  # SecurityContext for all containers
  ########################################################
  securityContext:
    runAsUser: 1000
    runAsGroup: 2000
    ...
    testSteps:
    - name: 01-init-example
      ...
      securityContext:
        runAsUser: ...
    ...
    finalStep:
      securityContext:
        runAsUser: ...
```

- Issue #167 — Disable testFramework by default. To enable, simply:

```
testFramework:
  enabled: true
  ...
```

Release 0.6.5 (July 4, 2021)

- Issue #163 — Add PingAuthorize and PingAuthorizePAP to helm charts.

  This includes the pre-release to PingAuthorize 8.3. It includes the necessary config for PingAuthorize and PingAuthorizePAP, even though there isn't a release for 2105. The current edge release is required to use the default server-profiles provided in the values.yaml. Once the global tag is changed to 2106 (over next few days) PingAuthorize will be default for use over PingDataGoverance. This will be tracked in a ticket released 2105.

  Example yaml to test PingAuthoize/PAP:

```
pingdataconsole:
  enabled: true

pingdirectory:
  enabled: true

pingauthorize:
  image:
    tag: 8.3.0.0-edge
  enabled: true

pingauthorizepap:
  enabled: true
```

Release 0.6.4 (July 1, 2021)

- Issue #158 — Increment default tag to 2105 Sidecars and initContainers are valuable for a multitude of reasons - log forwarding, metric exporting, backup jobs. Because of this they can also have many ways of being configured.

  Allow for defining three top level maps to provide details for:

  - sidecars — Defines sidecar containers to be run alongside product containers.
  - initContainers — Defines initContainers to be run before product containers.
  - volumes — Defines volumes used by sidecars, initContainers and product containers.

  Example definitions:

```
sidecars:
  pd-access-logger:
    name: pd-access-log-container
    image: pingidentity/pingtoolkit:2105
    volumeMounts:
      - mountPath: /tmp/pd-access-logs/
        name: pd-access-logs
        readOnly: false
  statsd-exporter:
    name: statsd-exporter
    image: prom/statsd-exporter:v0.14.1
    args:
    - "--statsd.mapping-config=/tmp/mapping/statsd-mapping.yml"
    - "--statsd.listen-udp=:8125"
    - "--web.listen-address=:9102"
    ports:
      - containerPort: 9102
        protocol: TCP
      - containerPort: 8125
        protocol: UDP

initContainers:
  init-1:
    name: 01-init
    image: pingidentity/pingtoolkit:2105
    command: ['sh', '-c', 'echo "Initing 1" && touch /tmp/pd-access-logs/init-1']
    volumeMounts:
      - mountPath: /tmp/pd-access-logs/
        name: pd-access-logs
        readOnly: false

volumes:
  pd-access-logs:
    emptyDir: {}
  statsd-mapping:
    configMap:
      name: statsd-config
      items:
        - key: config
          path: statsd-mapping.yml
```

And within the product (or global) definition, allow for inclusion of sidecars, initContainers and volumes. These must be available in the top-level sidecars:, initContainers:, and volumes:

- includeSidecars
- includeInitContainers — Run in order as listed in array
- includeVolumes

Example usages:

```
pingdirectory:
  ...
  includeSidecars:
    - pd-access-logger
  includeInitContainers:
    - init-1
  includeVolumes:
    - pd-access-logs

  volumeMounts:
    - mountPath: /opt/access-logs/
      name: pd-access-logs
```

Release 0.6.3 (June 21, 2021)

- Issue #154 — Increment default tag to `2105`.
- Issue #155 — Add clusterServiceName to product services with service clusters.

Release 0.6.2 (May 24, 2021)

- Issue #151 — Add support for Container LifeCycle Event Hooks.

  Adding the following to values.yaml:

  ```
  global:
    ##############################################################
    # container life handlers, allowing for lifecycle events such
    # as postStart and preStop events
    #
    # https://kubernetes.io/docs/tasks/configure-pod-container/attach-handler-lifecycle-event
    ##############################################################
    lifecyle: {}
    # Example
    # lifecyle:
    #   postStart:
    #     exec:
    #       command: ["/bin/sh", "-c", "echo Start Complete > /tmp/message"]
  ```

- General cleanup of values.yaml comments.

- Setting default `externalImages.pingtoolkit` tag to `2104`, and removing `edge` tag from `ldap-sdk-tools` image which will now default to same `global.image.tag` setting (currently 2104).

Release 0.6.1 (May 21, 2021)

- Issue #148 — Calculate checksum of `ConfigMaps` based on the data rather than entire `ConfigMap` file.

  This will only use the `ConfigMap.data` when creating checksums in workload rather than using the entire file. It will result in no checksum change when labels/annotations are the only thing changing. A good example is the helm chart version, which changes the label, but not data.

Release 0.6.0 (May 11, 2021)

- Changed default `global.image.pullPolicy` from `Always` to `IfNotPresent`.

  This is due to the fact that the `global.image.tag` is a non-floating tag. Once it is downloaded and present, it will not change. This small change will increase performance at startup as images are typically present when installing/updating releases.

  Simply set `global.image.pullPolicy=Always` to pull every time if needed.

- BETA 2 - Testing Framework supporting helm test command and associated `testFramework` values.

  Cleaned up the generation of resources honoring the `addReleaseNameToResource` setting.

Release 0.5.9 (May 10, 2021)

- BETA 1 - Testing Framework supporting helm test command and associated `testFramework` values.

  A testing framework is being created to allow for testing Ping Identity helm chart deployments using a `testFramework` set of values. This is currently in beta, with documentation to available soon. Expect that changes will be made to this work, until it's fully released with documentation.

Release 0.5.8 (May 6, 2021)

- Issue #141 — Fix `DNS_QUERY_LOCATION` on pingfederate-engine configmap.yaml

  Resolves an issue with the `DNS_QUERY_LOCATION` when pingfederate clustering is used for >1 pingfederate-engines

Release 0.5.7 (May 3, 2021)

- Issue #136 — ClusterIP Services port/targetPort be set to the containerPort

  Since the ClusterIP Services (aka Headless services) only provide access to the underlying container IP and port. The port, and by default `targetPort`, will be set to the `containerPort` value. The helm charts will start requiring the `containerPort` for any service where `clusterService:true` is set, otherwise it will fail with an error message.

- Issue #138 — Update `image.tag` to 2104 (April 2021)

Release 0.5.5 (April 29, 2021)

- Issue #133 — Change default pingdirectory values (container.resources.requests.cpu=50m and container.replicaCount=1)

  Setting the cpu request to 50m, will provide at last some reservation of CPU, so that if there are multiple nodes, it will better even out the load.

  Additionally, setting the `replicaCount` to 1 by default, as many cases in development, there isn't a great need to have multiple replicas. If this is the case, simply set `pingdirectory.container.replicaCount=2` or any number of replica's.

- Issue #132 - Adding PingDirectoryProxy to mix of products

Release 0.5.5

- Issue #126 — Unable to mount `secretVolume` and `configMapVolumes` simultaneously

  This is one additional fix to the the same thing fixed in 0.5.4. `volumeMounts:` had the same issue as `volumes:`. This completes and resolves issue #126.

Release 0.5.4

- Issue #126 — Unable to mount `secretVolume` and `configMapVolumes` simultaneously

  Due to the fact that `volumes:` is an array of items, `volumes:` usage with `secret` or `configMap` volumes exposed the issue that multiple `volumes:` entries were used, and only kept the last one. Fix included only using `volumes:` once. Note that the template will end up with a `volumes: null` if none are set (i.e. deployment with no Secret/ConfigMap volumes), but that is ok.

Release 0.5.3

- Issue #121 — Create `global-env-vars` hosts/ports for all products regardless if enabled

  The status of this config map is used to form the checksum for the products. This will ensure that a simple addition/deletion of a product from the deployed mix won't cause all products to be restarted.

- Issue #122 - Update `image.tag` to 2103 (March 2021)

  The image tag is modified to 2103. This includes:

  - Security Context on StatefulSets to include a fsGroup=9999 (same as gid)
  - Update the services ContainerPort to unprivileged ports (i.e. 636 -→ 1636)

Release 0.5.2

- Issue #113 — Default pingaccess-admin to StatefulSet

  In order to provide HA with a PingAccess cluster between admin/engine nodes, it is required that the PingAccess Admin deploy as a StatefulSet with persistence. Otherwise if the PingAccess Admin goes down, the engines would lose connectivity to that node and be unable to get further config updates and subsequently have to bounce and lose their web-session information.

The new default yaml

```
pingaccess-admin:
  workload:
    type: StatefulSet
```

- Issue #95 — Fix default serviceAccount in workload for vault

Fixed issue that was created in Issue 95 (using annotations to provide vault details) to pull serviceAccountName from the proper location in annotations.

```
  vault:
    hashicorp:
      annotations:
        serviceAccountName: vault-auth
```

- Issue #116 — Support Annotations at Workload Level.

Support annotations at the workload level. For workloads, adding `.spec.template.metadata`.

Example telegraf annotation:

```
pingfederate-engine:
  workload:
    annotations:
      telegraf.influxdata.com/class: app
```

would lead to:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app.kubernetes.io/instance: samir
    app.kubernetes.io/managed-by: Helm
    app.kubernetes.io/name: pingfederate-engine
    helm.sh/chart: ping-devops-0.5.1
  name: samir-pingfederate-engine
spec:
  replicas: 1
  selector:
    matchLabels:
      app.kubernetes.io/instance: samir
      app.kubernetes.io/name: pingfederate-engine
  strategy:
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 0
    type: RollingUpdate
  template:
    metadata:
      annotations:
        telegraf.influxdata.com/class: app
```

- Issue #117 — Bug - cluster service shouldn't use image name for service name.
- Issue #114 — Revamp vault.hashicorp.secrets value .yaml and support per path secret Detailed documentation on this can be found the Vault Configuration docs

Release 0.5.1

- Added back in the service name by default to the private cert generation pulled out of the previous release by accident.

  If the product was `pingaccess-admin` and release was `acme`, then the service name might be `acme-ping-access-admin`. This name by default will be added to the alternative hosts of the private certificate generation by default. Without this the pingaccess clustering will fail during setup.

Release 0.5.0

- Issue #103 - Provide ability to add additional alt-names/alt-ips to private cert generation

  Allow for a privateCert structure to contain optional arrays `additionalHosts` and `additionalIPs`:

```
pingaccess-admin:
  privateCert:
    generate: true
    additionalHosts:
    - pingaccess-admin.west-cluster.example.com
    - pa-admin.west-cluster.example.com
    additionalIPs:
    - 123.45.67.8
```

In addition, if the ingress for the product is enabled, the host(s) created for that ingress will also be added to the alt-names.

The above example (with an ingress) will create a cert used by pingaccess-admin containing:

```
    Certificate:
        Data:
            ...
        Signature Algorithm: sha256WithRSAEncryption
            Issuer: CN=pingaccess-admin
            ...
            X509v3 extensions:
                ...
                X509v3 Subject Alternative Name:
                    DNS:rel050-pa-pingaccess-admin.ping-devops.com. pingaccess-admin.west-cluster.example.com,
DNS:pa-admin.west-cluster.example.com, IP Address:123.45.67.8
```

Release 0.4.9

- Issue #104 — Update default global image tag to 2102 (Feb 2021)

  Update the default global image tag in base values.yaml and remove edge from example yamls.

Release 0.4.8

- Issue #100 — Change pingfederate-engine HPA to a default of disabled

  Changing the default value `pingfederate-engine.clustering.autoscaling.enabled=false`, since the default CPU Request is set to 0.

Release 0.4.7

- Issue #95 — Unable to set numerous Vault configuration options

  Updated ability to add any hashicorp.vault annotation to the workload. As part of this effort, the existing name/values have been deprecated, however will continue to work for a period of time.

  Updated details can be found in the Vault Config docs.

- Issue #97 — Add the ability to add annotations to all resources generated similar to current support for Labels. This will allow deployers to specify additional annotations at either the global and/or product level. An example of the values yaml would look like:

```
    global:
      annotations:
        app.ping-devops.com/test: test-name

    pingaccess-admin:
      annotations:
        app.pingaccess/version: v1234
```

Additional cleanup of Notes.txt outputting detail of deployment.

Release 0.4.6

Minor follow-up update to cpu/memory request/limit sizes for init containers.

Release 0.4.5

- Issue #89 — Update default workload resource cpu/memory request sizes.

  Updating defaults to create a usage better reflecting actual memory usage by product. And minimizing amount of CPU needed as testing generally utilizes very little. Of course, it is definitely recommended that production deployments specify amount of cpu and memory required and limited to.

  Current defaults are set to:

```
#-------------------------------------------------------------------------------------
# Ping DevOps
#
# Description: All Ping Identity product images with integration
#-------------------------------------------------------------------------------------
#
#           Product          Workload   cpu-R cpu-L mem-R mem-L  Ing
#    --------------------- ----------- ----- ----- ----- ----- -----
#  √ pingaccess-admin      deployment  0     2     1Gi   4Gi   false
#  √ pingaccess-engine     deployment  0     2     1Gi   4Gi   false
#  √ pingdataconsole       deployment  0     2     .5Gi  2Gi   false
#  √ pingdatagovernance    deployment  0     2     1.5Gi 4Gi   false
#  √ pingdatagovernancepap deployment  0     2     .75Gi 2Gi   false
#  √ pingdatasync          deployment  0     2     .75Gi 2Gi   false
#  √ pingdelegator         deployment  0     500m  32Mi  64Mi  false
#  √ pingdirectory         statefulset 0     2     2Gi   8Gi   false
#  √ pingfederate-admin    deployment  0     2     1Gi   4Gi   false
#  √ pingfederate-engine   deployment  0     2     1Gi   4Gi   false
#
#  √ ldap-sdk-tools        deployment  0     0     0     0     false
#  √ pd-replication-timing deployment  0     0     0     0     false
#
#-------------------------------------------------------------------------------------
```

Release 0.4.4

- Issue #80 — Add support for importing a secret containing license into the container. Adds ability to add secret and configMap data to a container via a VolumeMount. A good use of this practice - bringing product licenses into the container.

  Example of creating 3 volume mounts in container from secret and configMap

```
    pingfederate-admin
      secretVolumes:
        pingfederate-license:
          items:
            license: /opt/in/instance/server/default/conf/pingfederate.lic
            hello: /opt/in/instance/server/default/hello.txt

      configMapVolumes:
        pingfederate-props:
            items:
              pf-props: /opt/in/etc/pingfederate.properties
```

In this case, a secret (called pingfederate-license) and configMap (called pingfederate-props) will bring in a couple of key values (license, hello) and (pf-props) into the container as specific files. The results will looks like:

Example of kubectl describe of pingfederate-admin container

```
    Containers:
      pingfederate-admin:
        Mounts:
          /opt/in/etc/pingfederate.properties from pingfederate-props (ro,path="pingfederate.properties")
          /opt/in/instance/server/default/conf/pingfederate.lic from pingfederate-license
(ro,path="pingfederate.lic")
          /opt/in/instance/server/default/hello.txt from pingfederate-license (ro,path="hello.txt")
    Volumes:
      pingfederate-license:
        Type:        Secret (a volume populated by a Secret)
        SecretName:  pingfederate-license
        Optional:    false
      pingfederate-props:
        Type:      ConfigMap (a volume populated by a ConfigMap)
        Name:      pingfederate-props
        Optional:  false
```

Release 0.4.3

- Issue #83 — Remove old pingdirectory tag check when creating service-cluster. This caused issues when creating a pingdirectory deployment with most recent tags (tags other than edge or 2012).

Release 0.4.2

- Issue #79 — Adding support for product PingDataGovernance PAP
- Issue #78 — Adding support to provide affinity definition to the workload of a product.

  Example values.yaml to add podAntiAffinity to pingdirectory

```
pingdirectory:
  container:
    affinity:
      podAntiAffinity:
        # Add a hard requirement for each PD pod to be deployed to a different node
        requiredDuringSchedulingIgnoredDuringExecution:
        - labelSelector:
            matchExpressions:
            - key: app.kubernetes.io/name
              operator: In
              values:
              - pingdirectory
          topologyKey: "kubernetes.io/hostname"
        # Add a soft requirement for each PD pod to be deployed to a different AZ
        preferredDuringSchedulingIgnoredDuringExecution:
        - weight: 1
          podAffinityTerm:
            labelSelector:
              matchExpressions:
              - key: app.kubernetes.io/name
                operator: In
                values:
                - pingdirectory
            topologyKey: "failure-domain.beta.kubernetes.io/zone"
```

Release 0.4.1

- Change default image tag to `2101` (January 2021).
- Create private certs and keystore for use by images, only if the value `{product-name}.privateCert.generate=true`. Defaults are false.
    - Helm will generate the a `tls.crt` and `tls.key`, place it into a kubernetes secret called `{release-productname}-private-cert`.
    - Mount the secret into the image under `/run/secrets/private-cert`
    - An init container will pull the `tls.crt` and `tls.key` into a pkcs12 keystore and place it into a file `/run/secrets/private-keystore/keystore.env` that will be mounted into the running container.
    - When the container's hooks are running, it will source the environment variables in this `keystore.env`. The default variables set are:
        - PRIVATE_KEYSTORE_PIN={base64 random pin}
        - PRIVATE_KEYSTORE_TYPE=pkcs12
        - PRIVATE_KEYSTORE={pkcs12 keystore}

      ```
      yaml to generate a private cert/keystore for pingaccess-admin:
      pingaccess-admin:
        privateCert:
          generate: true
      Example of created /run/secrets/private-keystore/keystore.env
      ```

```
PRIVATE_KEYSTORE_PIN=nrZmV4XdfK....
PRIVATE_KEYSTORE_TYPE=pkcs12
PRIVATE_KEYSTORE=MIIJgQIBAzCCCUcGC....
```

- Added support for PingAccess clustering between pingaccess-admin and multiple pingaccess-engine containers.
  - See everything.yaml for example of deploying a PingAccess cluster using PingFederate/PingDirectory to authenticate
  - It is required to either:
    - generate the private cert (see above) with the value of pingaccess-admin.privateCert.generate=true or
    - provide your own cert secret called {release-productname}-private-cert containing a valid tls.crt and tls.key.
  - Enable both the pingaccess-admin and pingaccess-engine helm chart products
    - Example values to create a clustered pingaccess:

```
pingaccess-admin:
  enabled: true
  privateCert:
    generate: true
  envs:
    SERVER_PROFILE_URL: https://github.com/pingidentity/pingidentity-server-profiles.git
    SERVER_PROFILE_PATH: baseline/pingaccess
pingaccess-engine:
  enabled: true
  envs:
    SERVER_PROFILE_URL: https://github.com/pingidentity/pingidentity-server-profiles.git
    SERVER_PROFILE_PATH: baseline/pingaccess
pingfederate-admin:
  enabled: true
  envs:
    SERVER_PROFILE_URL: https://github.com/pingidentity/pingidentity-server-profiles.git
    SERVER_PROFILE_PATH: baseline/pingfederate
  container:
    waitFor:
      pingdirectory:
        service: ldaps
pingfederate-engine:
  enabled: true
  envs:
    SERVER_PROFILE_URL: https://github.com/pingidentity/pingidentity-server-profiles.git
    SERVER_PROFILE_PATH: baseline/pingfederate
pingdirectory:
  enabled: true
  envs:
    SERVER_PROFILE_URL: https://github.com/pingidentity/pingidentity-server-profiles.git
    SERVER_PROFILE_PATH: baseline/pingdirectory
```

Release 0.4.0

- Support availability of PingDirectory pods through the cluster headless kubernetes service. Allows for PingDirectory nodes to find one another during the replication enable/init process.

Adds following to pingdirectory-cluster:

```
metadata:
  annotations:
    service.alpha.kubernetes.io/tolerate-unready-endpoints: "true"
spec:
  publishNotReadyAddresses: true
```

Release 0.3.9

- Fixed the default wait-for service name on pingfederate-engine (admin -→ https).
- Changed default on readiness command to check for readiness every 5 seconds rather than 30. This allows for availability on some services, such as PingFederate which is normally ready in 30 sec.

Release 0.3.8

- Issue #56 — Improved Default Naming on Global vars - PORTs
- Issue #56 — Improved Default Naming on Global vars - PORTs
- In Release 0.3.6, global-env-vars were created for PORTS. The naming structure used was complex and difficult, primarily because a product can have several ports open on a particular private and public host. The format will be more consistent as defined by the following:

`{product-short-code with type}_{public or private}_{hostname or port}{_service if port}`

An example with PD might look like (note the service names of `https` and `data-api`):

```
PD_ENGINE_PUBLIC_PORT_HTTPS: 443
PD_ENGINE_PUBLIC_PORT_DATA_API: 1443

PD_ENGINE_PRIVATE_PORT_HTTPS: 443
PD_ENGINE_PRIVATE_PORT_DATA_API: 8443
```

- Issue #62 — When creating configMapRef's, take into account the proper release name to include

  ConfigMapRef names in workloads were not consistent with the ConfigMaps created by default when taking into account the `addReleaseNameToResource` setting of prepend, append or none. This fixes that issue ensuring that config maps are consistent.

- Added global-env private/public host/port for PingDataConsole, which was missing.
- Changed the default pingfederate-admin `admin` service name to `https` to reduce confusion.
- Changed the default pingfederate-engine `engine` service name to `https` to reduce confusion.

Release 0.3.7

- Fixes issue with service -vs- ingress name on creation of ingress to service mapping. Resolves issue #57.

Release 0.3.6

- Cleaning up and making services/ingresses easier to use together. Incorporating all the ports used in both a service and ingress into the same location of the Service Configuration.

  The example below shows a container/service/ingress and how to specify the ports at each level.

  - `containerPort` → Replaces `targetPort`
  - `servicePort` → Replaces `port`
  - `ingressPort` → New entry

```
services:
  api:
    containerPort: 8443 <--- changed from targetPort
    servicePort: 1443   <--- changed from port
    ingressPort: 443    <--- new.  moved from ingress
    dataService: true
  data-api:
    containerPort: 9443 <--- changed from targetPort
    servicePort: 2443   <--- changed from port
    ingressPort: 2443   <--- new.  moved from ingress
    dataService: true
ingress:
  hosts:
    - host: pingdirectory.example.com
      paths:
      - path: /api
        backend:
          serviceName: api        <--- changed from servicePort
      - path: /directory/v1
        backend:
          serviceName: data-api   <--- changed from servicePort
```

  Additionally, `global-env-vars` will be created for each of these ports. If the name of the product is `PROD`, the the following ports would be created:

```
PROD_API_PRIVATE_PORT="1443"          # This is the servicePort
PROD_API_PUBLIC_PORT="443"            # This is the ingressPort
PROD_DATA_API_PRIVATE_PORT="2443"
PROD_DATA_API_PUBLIC_PORT="2443"
```

- Fixed missing `USER_BASE_DN` setting in simple-sync.yaml example.

Release 0.3.5

- Allowing config values to determine use of init containers to wait-for other chart products. For each product, you can now provide a `waitFor` structure providing the name and service that should be waited on before the running container con continue. This will basically inject an initContainer using the PingToolkit wait-for utility until it can `nc host:port` before continuing.

PingFederate Admin waiting on pingdirectory ldaps service to be available:

```
pingfederate-admin:
  container:
    waitFor:
      pingdirectory:
        service: ldaps
      pingdatagovernance:
        service: https
```

- By default, the `pingfederate-engine` will waitFor `pingfederate-admin` before it starts.

Release 0.3.4

- Adding init container to PingFederate Admin to wait-for PingDirectory's LDAPs port if the pingdirectory.enabled=true. This fixes an issue that keeps PingFederate Admin from starting when it's dependent on PingDirectory. In the case that PingFederate isn't dependent on PingDirectory and it is still enabled, it will simply delay the start time of PingFederate admin. A future version will allow for specifying a list of services to wait-for so this can be turned off/on by deployer.
- Moved the securityContext settings added to release 0.3.3 from the container to the workload, as that is the proper place to use them. Required for use of `fsGroup` setting.

Release 0.3.3

- Adding the ability for a deployer to add a securityContext to the containers. Currently, there are warning messages in the images when an outside-in pattern is used (i.e. securityContext is set). Also, many of the default ports require privileged access, so care should be taken along with testing to ensure the containers start up fine. Additional, one should not change the security context when doing and upgrade or using a PCV from a previous deployment.

  An example securityContext that can be used might look like:

```
global:
  container:
    securityContext:
      allowPrivilegeEscalation: false
      capabilities:
        drop:
        - ALL
      runAsGroup: 1000
      runAsNonRoot: true
      runAsUser: 100
```

  By default, the values.yaml in the chart will set the securityContext to empty:

```
global:
  container:
    securityContext: {}
```

Release 0.3.2

- Replaced init container on pingfederate-engine to use pingtoolkit rather than 3rd party curlimage. Additionally added resource constraints and security context to this init container.
- Removed hardcoded SERVER_PROFILE_BRANCH set to master, relying on git repo default branch
- Cleaned up pingdelegator values. public hostnames for pingfederate and pingdirectory built based off of ingress hostnames, part of `{release-name}-global-env-vars` configmap.
- Removed default nginx annotations of ingress resources. If an nginx controller is used for ingress, the following ingress annotations should be included:

> **Warning**
>
> By removing the following annotations from the default, use of current config values will result in no ingress being set. You must add these in via your .yaml file or via separate --set settings.
>
> ```
> global:
>   ingress:
>     annotations:
>       nginx.ingress.kubernetes.io/backend-protocol: "HTTPS"
>       kubernetes.io/ingress.class: "nginx-public"
> ```

Release 0.3.1

- Added container envFrom for `{release-name}-env-vars` back as optional. Fixes breaking change from 0.2.8 to 0.2.9 for those that used this configmap.
- Added ability for deployer to add their own envFrom's via their values.yaml. An example (adding an optional configmap/secrets to all products). Just change global to the name of the product to only have that product use the references.

```
global:
  container:
    envFrom:
    - configMapRef:
        name: my-killer-configmap
        optional: true
    - secretRef:
        name: my-killer-secrets
        optional: true
```

Release 0.3.0

- Consolidate deployment/stateful set templates to a single workload template.
- Changes to values.yaml:
  - Created a workload map under global (see below)
  - Moved old deployment information under workload

- Moved old statefulSet information under workload
- Updated `pingfederate-admin` to reflect new workload
- Updated `pingdirectory` to reflect new workload
- Allows for any product to be run as a deployment or statefulSet

Warning

Using `workload.type=StatefulSet` will create `pvc` resources and allow for persistence on restarts of containers. This is helpful during development. Be aware that the `pvc` resources will need to be deleted to startup a fresh copy of the product images.

```
global: workload: type: Deployment # Can be Deployment or StatefulSet (see warning above)

    deployment:
      strategy:
        type: RollingUpdate # Can be RollingUpdate or Recreate
        rollingUpdate:
          maxSurge: 1
          maxUnavailable: 0

    statefulSet:
      partition: 0          # Used for canary testing if n>0

      persistentvolume:
        enabled: true
        ##############################################################
        # For every volume defined in the volumes list, 3 items will be
        # created in the StatefulSet
        #   1. container.volumeMounts - name and mountPath
        #   2. template.spec.volume - name and persistentVolumeClaim.claimName
        #   3. spec.volumeClaimTemplates - persistentVolumeClaim
        #
        # https://kubernetes.io/docs/concepts/storage/persistent-volumes/
        ##############################################################
        volumes:
          out-dir:
            mountPath: /opt/out
            persistentVolumeClaim:
              accessModes:
              - ReadWriteOnce
              storageClassName:
              resources:
                requests:
                  storage: 4Gi
```

- Renamed template files in pinglib from .yaml to .tpl
- Added `terminationGracePeriodSeconds` to container to support setting in values
- Added `serviceAccountName` to vault.hashicorp to specify to the container what service account can be used to authenticate to the Hashicorp Vault Injector