# ESTA Tekton Self-Managed Migration

ESTA Tekton can now be installed individually using a prepared Helm chart which is maintained and released by ESTA. The preparation and use of this Helm chart is documented in the README of the [esta-tekton-helm](https://code.sbb.ch/projects/KD_ESTA/repos/esta-tekton-helm/browse) source repository (see section "How to use").

If you're already using ESTA Tekton as a managed service (ordered via the CLEW Self Service Portal), here are the steps required to migrate your Tekton instance into your own self-managed environment:

* 1 [Backup the Tekton Results Database](#ESTATektonSelfManagedMigration-Backupth)
* 2 [(A) Migrate the OpenShift Namespace](#ESTATektonSelfManagedMigration-(A)Migra)
* 3 [(B) Create a new OpenShift Namespace](#ESTATektonSelfManagedMigration-(B)Creat)
* 4 [Shut down the managed ESTA Tekton instance](#ESTATektonSelfManagedMigration-Shutdown)
* 5 [Install ESTA Tekton via ArgoCD](#ESTATektonSelfManagedMigration-InstallE)
* 6 [Restore the Tekton Results Database](#ESTATektonSelfManagedMigration-Restoret)

## Backup the Tekton Results Database

For new Tekton instances, a new database schema for storing pipeline runs will be created. In order to preserve the build history, we recommend to dump the existing database and later import the records into the new schema.

1. Get the database credentials from the secret named "tekton-results-postgres" in the existing Tekton namespace
2. Use [pg\_dump](https://www.postgresql.org/docs/current/app-pgdump.html) to copy all data into a local file

Here's a shell script that performs these actions:

**backup-tekton-db.sh**

# 1. log in to the cluster that hosts your current Tekton namespace

# oc login ....

export TEKTON\_NAMESPACE="<your-tekton-namespace>"

export POSTGRES\_SCHEMA=$(echo "$TEKTON\_NAMESPACE" | sed 's/-/\_/g')

for key in DB\_HOST POSTGRES\_DB POSTGRES\_USER POSTGRES\_PASSWORD; do

export $key=$(oc get secret tekton-results-postgres -n $TEKTON\_NAMESPACE -o jsonpath="{.data.$key}" | base64 -d);

done

PGPASSWORD="$POSTGRES\_PASSWORD" pg\_dump -U $POSTGRES\_USER -h $DB\_HOST -p 5432 -n $POSTGRES\_SCHEMA --no-privileges --no-comments --data-only $POSTGRES\_DB > tekton-results-dump.sql

## (A) Migrate the OpenShift Namespace

For a smooth transition it's best to keep the existing namespace but simply replace the deployment of ESTA Tekton. The following steps are required:

1. Change the label clew.sbb.ch/project-type: build to the value "build-self"
2. Add the label iam.sbb.ch/appregistration: "true"
3. Remove sensitive secrets: azure-ad-client-credentials, esta-owasp-db-secret, esta-tekton-smtp-auth, esta-tekton-sonar-api-key
4. Grant admin rights to the new owner/customer

**admin-role-binding.yaml** Quelle erweitern

apiVersion: rbac.authorization.k8s.io/v1

kind: RoleBinding

metadata:

name: admin-uXXXXXX

subjects:

- kind: User

apiGroup: rbac.authorization.k8s.io

name: uXXXXXX

roleRef:

apiGroup: rbac.authorization.k8s.io

kind: ClusterRole

name: admin

IMPORTANT

The label clew.sbb.ch/project-type can only be changed by a cluster admin. Reach out to the CON team to get this done.

## (B) Create a new OpenShift Namespace

Order a new OpenShift namespace via the [CLEW Self Service Portal](https://self.sbb-cloud.net/tools/openshift/standalone). Tekton namespaces **can only be created on the clusters aws01t and aws02t**.

In order to keep the URL to your Tekton instance unchanged, we recommend to choose the same name as the existing namespace (e.g. clew-tekton) but select another cluster. So if your current namespace is on aws01t, create the new one on aws02t and vice versa. Of course, you can choose any name you want but the URL pointing to your Tekton instance will contain the namespace name.

IMPORTANT

At the time of writing, some changes to the namespace template need to be applied manually be the Container Team. Please open a CLEW Support issue with the component "OpenShift" to get the default limit ranges and resource quotas removed from your Tekton namespace and to get the label clew.sbb.ch/project-type: build-self added.

## Shut down the managed ESTA Tekton instance

1. Remove the according application from the [ESTA ArgoCD Git repository](https://code.sbb.ch/projects/KD_ESTA/repos/esta-apps-argocd/browse).
2. Sync ArgoCD (should be done automatically)

## Install ESTA Tekton via ArgoCD

1. Follow the "How to use" section in the [README](https://code.sbb.ch/projects/KD_ESTA/repos/esta-tekton-helm/browse/README.md) and create a new application in your in your ArgoCD Git repository with two values files providing all the mandatory chart values listed in the README. One containing the public settings for your setup and one containing the secret values like client secrets, usernames, passwords, etc. The latter one should be encrypted with [SOPS](file:///C:\display\CLEW\ArgoCD+Setup+and+Configuration#ArgoCDSetupandConfiguration-ManageSecretValueswithSOPSandGPG) before checking it in to Git.
2. applications/esta-tekton/
3. - Chart.yaml
4. - values.yaml

- secrets.yaml

See [this example](https://code.sbb.ch/projects/KD_ESTA/repos/esta-apps-argocd/browse/applications/esta-tekton-helm) for reference.

1. Add an application template to your app-of-apps chart for the cluster where Tekton is supposed to be installed ([example](https://code.sbb.ch/projects/KD_ESTA/repos/esta-apps-argocd/browse/clusters/aws02t/applications/templates/esta-tekton/u233658-dev-tekton.yaml))
2. Permit your ArgoCD instance to manage the new Tekton namespace (see [Namespace management](file:///C:\display\CLEW\ArgoCD+Setup+and+Configuration#ArgoCDSetupandConfiguration-PermitArgoCDtodeploytoothernamespaces))
3. Sync your app-of-apps chart in ArgoCD and let the deployment start

## Restore the Tekton Results Database

Similar to backing up the database, we use the (new) credentials from the "tekton-results-postgres" secret to connect to the new database and then use [pgsql](https://www.postgresql.org/docs/current/app-psql.html) to import the backupped records.

**restore-tekton-results.sh**

# 1. log in to the cluster that hosts your current Tekton namespace

# oc login ....

export OLD\_TEKTON\_NAMESPACE=<your-tekton-namespace>

export NEW\_TEKTON\_NAMESPACE=$OLD\_TEKTON\_NAMESPACE

export OLD\_POSTGRES\_SCHEMA=$(echo "$OLD\_TEKTON\_NAMESPACE" | sed 's/-/\_/g')

export NEW\_NAMESPACE\_SNAKE=$(echo "$NEW\_TEKTON\_NAMESPACE" | sed 's/-/\_/g')

export NEW\_POSTGRES\_SCHEMA="${NEW\_NAMESPACE\_SNAKE}\_<cluster>"

# migrate DB dump to be restored into new schema

echo "TRUNCATE TABLE $NEW\_POSTGRES\_SCHEMA.results CASCADE;" > tekton-results-dump.sql.new

cat tekton-results-dump.sql | sed "s/$OLD\_POSTGRES\_SCHEMA./$NEW\_POSTGRES\_SCHEMA./" >> tekton-results-dump.sql.new

for key in DB\_HOST POSTGRES\_DB POSTGRES\_USER POSTGRES\_PASSWORD; do

export $key=$(oc get secret tekton-results-postgres -n $NEW\_TEKTON\_NAMESPACE -o jsonpath="{.data.$key}" | base64 -d);

done

PGPASSWORD="$POSTGRES\_PASSWORD" psql -U $POSTGRES\_USER -h $DB\_HOST -p 5432 -d $POSTGRES\_DB -f tekton-results-dump.sql.new

Finally, restart the controller to pick-up the new records from the database:

oc rollout resart deployment esta-tekton-pipeline-controller -n <your-tekton-namespace>