K8s setup

Update aks core infra

===

- Update enVariables.env
- Update Secrets

Principle secret is different for different applications:
Jfrog cert remains same for applications
Docker cert remains same for applications

kubectl create secret docker-registry <my-registry>
--docker-server=<myserver.example.com> --docker-username=<user> --
docker-password=<password>
--docker-email=<fake@example.com> --output yaml --dry-run=client -n
<some-namespace> |

kubectl create secret docker-registry colesgroup-cths-nonprod-docker-virtual
--docker-server=http://colesgroup-cths-nonprod-docker-virtual.jfrog.io
--docker-username=@coles.com.au --docker-password= --output yaml --dry-
run=client
-n cths-dev | kubeseal --cert AKS-DGXP-DGKB-NONPROD-AUE.cer
--scope namespace-wide > colesgroup-cths-nonprod-docker-virtual-sealed-
secret-file.yaml

kubectl create secret docker-registry colesgroup-waos-sl-nonprod-docker-
virtual
--docker-server=colesgroup-waos-nonprod-docker-virtual.jfrog.io
--docker-username=vikas.kaushik@coles.com.au --docker-
password=rain4koh&d4J --output yaml --dry-run=client
-n cths-dev | kubeseal --cert AKS-DGXP-DGKB-NONPROD-AUE.cer
--scope namespace-wide > colesgroup-waos-nonprod-docker-virtual-
sealedsecret.yaml

kubectl create secret docker-registry colesgroup-waos-sl-nonprod-docker-virtual --docker-server=colesgroup-waos-nonprod-docker-virtual.jfrog.io —docker-username=vikas.kaushik@coles.com.au --docker-password=rain4koh&d4J --output yaml --dry-run=client -n waos-dev | kubeseal --cert AKS-DGXP-DGKB-NONPROD-AUE.cer --scope namespace-wide > dev-colesgroup-waos-nonprod-docker-virtual-sealedsecret.yaml

kubectl create secret docker-registry colesgroup-waos-sl-nonprod-docker-virtual --docker-server=colesgroup-waos-nonprod-docker-virtual.jfrog.io —docker-username=vikas.kaushik@coles.com.au --docker-password=rain4koh&d4J --output yaml --dry-run=client -n waos-test | kubeseal --cert AKS-DGXP-DGKB-NONPROD-AUE.cer --scope namespace-wide > test-colesgroup-waos-nonprod-docker-virtual-sealedsecret.yaml

kubectl create secret docker-registry colesgroup-waos-sl-nonprod-docker-virtual --docker-server=colesgroup-waos-nonprod-docker-virtual.jfrog.io —docker-username=vikas.kaushik@coles.com.au --docker-password=rain4koh&d4J --output yaml --dry-run=client -n waos-svt | kubeseal --cert AKS-DGXP-DGKB-NONPROD-AUE.cer --scope namespace-wide > svt-colesgroup-waos-nonprod-docker-virtual-sealedsecret.yaml

Download certificate and decrypt
Example :

cat waos.svt.k8s.dgxp.aue.azr.cmltd.net.au.key|openssl rsa -out unencrypted.key

kubectl create secret tls waos-sl-tls-secret-dev --key ./unencrypted.key --cert ./waos.svt.k8s.dgxp.aue.azr.cmltd.net.au.crt -n waos-dev --dry-run=client --output yaml | kubeseal --cert AKS-DGXP-DGKB-NONPROD-AUE.cer > dev.k8s.dgxp.aue.azr.cmltd.net.au.yaml

kubectl create secret tls waos-sl-tls-secret-test --key ./unencrypted.key --cert ./waos.svt.k8s.dgxp.aue.azr.cmltd.net.au.crt -n waos-test --dry-run=client --output yaml | kubeseal --cert AKS-DGXP-DGKB-NONPROD-AUE.cer > test.k8s.dgxp.aue.azr.cmltd.net.au.yaml

kubectl create secret tls waos-sl-tls-secret-svt --key ./unencrypted.key --cert ./waos.svt.k8s.dgxp.aue.azr.cmltd.net.au.crt -n waos-svt --dry-run=client --output yaml | kubeseal --cert AKS-DGXP-DGKB-NONPROD-AUE.cer > svt.k8s.dgxp.aue.azr.cmltd.net.au.yaml

One more example for MAOA-scan-and-go

Download cert maoa-tls-secret-nonprod from venafi

Then decrypt the key and create an unencrypted.key

cat maoa.svt.k8s.dgxp.aue.azr.cmltd.net.au.key |openssl rsa -out unencrypted.key

kubectl create secret tls maoa-sg-tls-secret-dev --key ./unencrypted.key --cert ./maoa.svt.k8s.dgxp.aue.azr.cmltd.net.au.crt -n maoa-dev --dry-run=client --output yaml | kubeseal --cert ../AKS-DGXP-DGKB-NONPROD-AUE.cer > dev-k8s.dgxp.aue.azr.cmltd.net.au.yaml

kubectl create secret tls maoa-sg-tls-secret-test --key ./unencrypted.key --cert ./maoa.svt.k8s.dgxp.aue.azr.cmltd.net.au.crt -n maoa-test --dry-run=client --output yaml | kubeseal --cert ../AKS-DGXP-DGKB-NONPROD-AUE.cer > test-k8s.dgxp.aue.azr.cmltd.net.au.yaml

kubectl create secret tls maoa-sg-tls-secret-svt --key ./unencrypted.key --cert ./maoa.svt.k8s.dgxp.aue.azr.cmltd.net.au.crt -n maoa-svt --dry-run=client --output yaml | kubeseal --cert ../AKS-DGXP-DGKB-NONPROD-AUE.cer > svt-k8s.dgxp.aue.azr.cmltd.net.au.yaml

Create service principal secret

kubectl create secret generic coles-app-sg-secrets --from-file=credentials.properties=[text file contains all secrets] -n maoa-test --dry-run=client -o yaml | kubeseal --cert ../AKS-DGXP-DGKB-NONPROD-AUE.cer --scope namespace-wide > dev-sg-nonprod-service-principle-secret.yaml

SPN-Coles-DEV-DigitalExperience-WAOS-SLS-ID
ba0eb7e5-8040-4f02-95cb-a8eaf90bdefb

SPN-Coles-DEV-DigitalExperience-WAOS-SLS-SECRET
cW5XhnGDY#J:>_gE}:}Q

==

Kubectl rollout undo deployment/dep-name

To find the difference before/after run kubectl get rs

```
import { check } from 'k6';
import http from 'k6/http';

const url = 'https://waos.dev.k8s.dgxp.aue.azr.cmltd.net.au/web-app-bff/
health'

export default function () {
  const res = http.get(url);
  check(res, {
    'is status 200': (r) => r.status === 200,
  });
}
=====
```