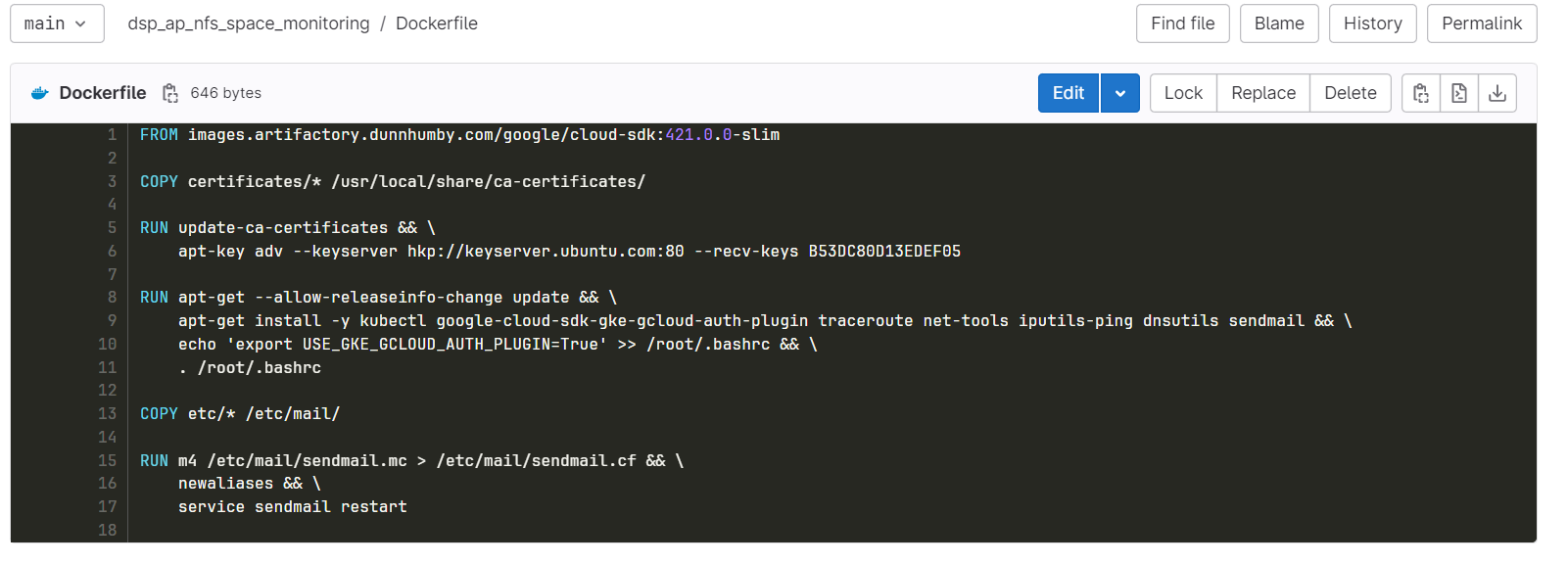
Dsp-ap-nfs-space-monitoring



A screenshot of a computer

Description automatically generated with medium confidence

Daily Standup Bot

Daily-standup-bot.sh

#!/bin/bash

set -xeuo pipefail

TODAY=$(date +%Y-%m-%d)

POST\_BODY='{title: "Standup '"$TODAY"'", text: "### \*[This message was generated from Gitlab CI]('"${CI\_JOB\_URL-blank}"')\* \n\n ## Current Focus \n\n - E.g., Incident, jira ticket, something else \n\n ## Blockers / Help / Something to share \n\n - E.g., Waiting for ServiceNow team to give me API access"}'

curl \

-H 'Content-Type: application/json' \

-d "$POST\_BODY" "$WEBHOOK\_URL"

A screenshot of a computer

Description automatically generated

Digicert expiring certs notification

Alert\_expiring\_certs.sh

#!/bin/bash

set -xuo pipefail

TODAY=$(date +%Y-%m-%d)

NINETY\_DAYS\_FROM\_TODAY=$(date -d "90 days" +%Y-%m-%d)

FORTEEN\_DAYS\_AGO\_FROM\_TODAY=$(date -d "14 days ago" +%Y-%m-%d)

curl "https://www.digicert.com/services/v2/order/certificate?filters\[status\]=issued&filters\[valid\_till\]=$TODAY...$NINETY\_DAYS\_FROM\_TODAY" \

-H "Content-Type: application/json" \

-H "X-DC-DEVKEY: $DIGICERT\_API\_KEY" \

| jq -r '[.orders[]] | sort\_by(.certificate.days\_remaining) | .[] | ["- [\(.certificate.common\_name)](https://www.digicert.com/secure/orders/\(.id)) expires on \(.order\_valid\_till) (in \(.certificate.days\_remaining) days)"] | @tsv' > expiring\_certs.tsv

curl "https://www.digicert.com/services/v2/order/certificate?filters\[status\]=issued&filters\[valid\_till\]=$FORTEEN\_DAYS\_AGO\_FROM\_TODAY...$TODAY" \

-H "Content-Type: application/json" \

-H "X-DC-DEVKEY: $DIGICERT\_API\_KEY" \

| jq -r '[.orders[]] | sort\_by(.certificate.days\_remaining) | .[] | ["- [\(.certificate.common\_name)](https://www.digicert.com/secure/orders/\(.id)) expired on \(.order\_valid\_till)"] | @tsv' > expired\_in\_past\_14\_days\_certs.tsv

NUMBER\_OF\_CERTS=$( wc -l < expiring\_certs.tsv )

NUMBER\_OF\_CERTS\_EXPIRED\_INPAST\_14\_DAYS=$( wc -l < expired\_in\_past\_14\_days\_certs.tsv )

echo "Certs Expired in past 14 days" "$NUMBER\_OF\_CERTS\_EXPIRED\_INPAST\_14\_DAYS"

POST\_BODY=$(jq -sR '{title: "'"$NUMBER\_OF\_CERTS"' Digicert certificates expiring before '"$NINETY\_DAYS\_FROM\_TODAY"'", text: "### \*[This message was generated from Gitlab CI]('"$CI\_JOB\_URL"')\* \n\n \(.)"}' < expiring\_certs.tsv )

curl \

-H 'Content-Type: application/json' \

-d "$POST\_BODY" "$WEBHOOK\_URL"

if [ -z "$NUMBER\_OF\_CERTS\_EXPIRED\_INPAST\_14\_DAYS"] || [ "$NUMBER\_OF\_CERTS\_EXPIRED\_INPAST\_14\_DAYS" == 0] ; then

POST\_BODY\_EXPIRED\_CERTS='{title: "No Certs Expired in past 14 days"}'

else

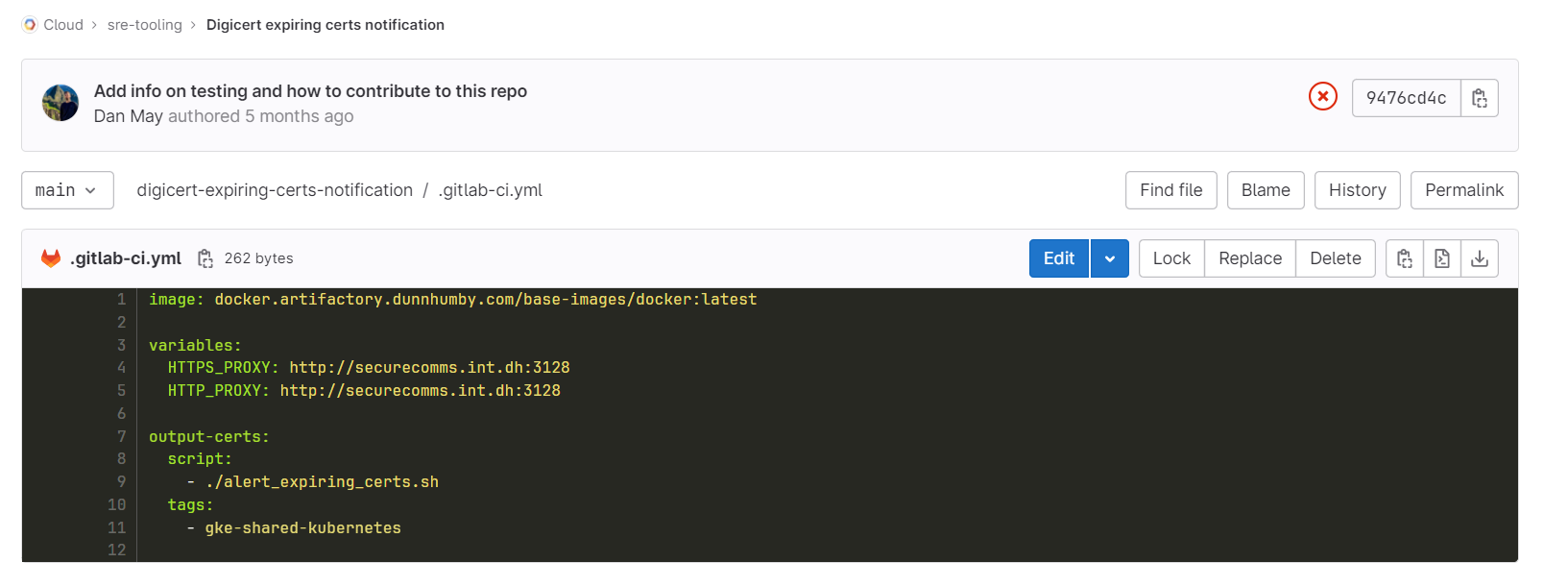
POST\_BODY\_EXPIRED\_CERTS=$(jq -sR '{title: "'"$NUMBER\_OF\_CERTS\_EXPIRED\_INPAST\_14\_DAYS"' Digicert certificates expired in past 14 days", text: "### \*[This message was generated from Gitlab CI]('"$CI\_JOB\_URL"')\* \n\n \(.)"}' < expired\_in\_past\_14\_days\_certs.tsv)

fi

curl \

-H 'Content-Type: application/json' \

-d "$POST\_BODY\_EXPIRED\_CERTS" "$WEBHOOK\_URL"



Internal cert renewal

Download\_pem.sh

#!/bin/bash

set -xeuo pipefail

ORDER\_ID=$1

curl -s "https://www.digicert.com/services/v2/certificate/download/order/$ORDER\_ID/format/pem\_noroot" \

-H 'Content-Type: application/json' \

-H "X-DC-DEVKEY: $DIGICERT\_API\_KEY"

Get\_core\_config.sh

#!/usr/bin/env bash

STACK\_NAME=$1

if [ "$#" -lt 1 ]; then

echo "Please provide STACK NAME as the first argument"

exit 1

fi

echo "Copying config from gs://${STACK\_NAME}-config"

gsutil -q cat "gs://${STACK\_NAME}-config/core.json" | jq . > core.json

gsutil -q cat "gs://${STACK\_NAME}-config/k8s.json" | jq . > k8s.json

export PROJECT=$(jq .project core.json -r)

export docker\_repo=$(jq .docker\_repo core.json -r)

export CLUSTER\_DNS=$(jq .cluster\_dns k8s.json -r)

export CLUSTER\_NAME=$(jq .cluster\_name k8s.json -r)

export REGION=$(jq .region core.json -r)

export ZONE=$(jq .zone core.json -r)

export WEBUSERPROXY\_DOMAIN=$(jq .webuserproxy\_domain core.json -r)

export DEPLOYMENT\_PREFIX=$(jq .deployment\_prefix core.json -r)

renew\_cert.sh

#!/bin/bash

set -xeuo pipefail

ORDER\_ID=$1

# Constants

ORG\_ID=350086 # dunnhumby Ltd

CONTAINER\_ID=394398 # CloudOperations

REPO\_URL="https://dhgitlab.dunnhumby.co.uk/Cloud/sre-tooling/internal-cert-renewal"

APACHE\_SERVER\_PLATFORM\_ID=2

EXPIRING\_ORDER\_DETAILS=$( curl -s "https://www.digicert.com/services/v2/order/certificate/$ORDER\_ID" \

-H "Content-Type: application/json" \

-H "X-DC-DEVKEY: $DIGICERT\_API\_KEY" )

CSR=$( echo $EXPIRING\_ORDER\_DETAILS | jq '.certificate.csr' )

CN=$( echo $EXPIRING\_ORDER\_DETAILS | jq '.certificate.common\_name' )

DNS=$( echo $EXPIRING\_ORDER\_DETAILS | jq '.certificate.dns\_names' )

EXPIRES\_AT=$( echo $EXPIRING\_ORDER\_DETAILS | jq '.certificate.valid\_till' )

IS\_RENEWED=$( echo $EXPIRING\_ORDER\_DETAILS | jq '.is\_renewed' )

WAS\_RENEWED\_BY\_ORDER\_ID=$( echo $EXPIRING\_ORDER\_DETAILS | jq '.was\_renewed\_by\_order\_id' )

if [[ "$IS\_RENEWED" == "true" ]]; then

echo "This order is not the latest one, maybe someone has already tried to renew this certificate???"

echo "Found a more recent order with id $WAS\_RENEWED\_BY\_ORDER\_ID. Please check if this certificate still requires renewal by looking at the new order."

echo "Aborting renewal process as a more recent order was found!"

exit 1

fi

RENEWED\_ORDER\_DETAILS=$( curl -s -X POST \

https://www.digicert.com/services/v2/order/certificate/ssl\_securesite\_flex \

-H "Content-Type: application/json" \

-H "X-DC-DEVKEY: $DIGICERT\_API\_KEY" \

-d '{

"certificate": {

"common\_name": '"$CN"',

"dns\_names": '"$DNS"',

"csr": '"$CSR"',

"signature\_hash": "sha256",

"server\_platform": {

"id": '$APACHE\_SERVER\_PLATFORM\_ID'

}

},

"comments": "Renewed by '$REPO\_URL'",

"container": {

"id": '$CONTAINER\_ID'

},

"custom\_renewal\_message": "Renewed by '$REPO\_URL'",

"organization": {

"id": '$ORG\_ID'

},

"order\_validity": {

"years": 1

},

"renewal\_of\_order\_id": '$ORDER\_ID'

}' )

RENEWED\_ORDER\_ID=$( echo $RENEWED\_ORDER\_DETAILS | jq '.id' )

export RENEWED\_ORDER\_ID=$RENEWED\_ORDER\_ID

export EXPIRES\_AT=$EXPIRES\_AT

revoke.sh

#!/bin/bash

set -xeuo pipefail

curl --request PUT "https://www.digicert.com/services/v2/order/certificate/$ORDER\_ID/revoke" \

-H 'Content-Type: application/json' \

-H "X-DC-DEVKEY: $DIGICERT\_API\_KEY" \

--data-raw '{

"skip\_approval": true

}'

#!/usr/bin/env bash

for i in "$@"

do

case $i in

--region=\*)

REGION="${i#\*=}"

shift # past argument=value

;;

--project=\*)

PROJECT="${i#\*=}"

shift # past argument=value

;;

--cluster\_name=\*)

CLUSTER\_NAME="${i#\*=}"

shift # past argument=value

;;

--master\_custom\_dns=\*)

MASTER\_CUSTOM\_DNS="${i#\*=}"

shift # past argument=value

;;

--default)

DEFAULT=YES

shift # past argument with no value

;;

\*)

# unknown option

;;

esac

done

KUBECONFIG\_CLUSTER=gke\_"$PROJECT"\_"$REGION"\_"$CLUSTER\_NAME"

echo "Manipulating kubeconfig entry using cluster: $KUBECONFIG\_CLUSTER"

# We would like to communicate to the GKE API server, which is exposed on a public IP address.

# We can't do this because these public IP addresses are no whitelisted by the hybrid securecomms instance,

# But .dunnhumby.cloud \*is\* whitelisted (and securecomms can do DNS resolution)

# So, we create a Cloud DNS A record with the public address of the GKE API server

# We also need to skip TLS because the certificate that is presented by the GKE APi server is only valid for:

# kubernetes, kubernetes.default, kubernetes.default.svc, kubernetes.default.svc.cluster.local, <API\_PUBLIC\_IP>, <API\_INTERNAL\_IP>

gcloud config set component\_manager/disable\_update\_check true

gcloud container clusters get-credentials --project="${PROJECT}" --region="${REGION}" "${CLUSTER\_NAME}"

kubectl config set "clusters.$KUBECONFIG\_CLUSTER.server" "https://$MASTER\_CUSTOM\_DNS"

kubectl config set "clusters.$KUBECONFIG\_CLUSTER.insecure-skip-tls-verify" true

kubectl config set "clusters.$KUBECONFIG\_CLUSTER.proxy-url" "http://securecomms.int.dh:3128"

kubectl config unset "clusters.$KUBECONFIG\_CLUSTER.certificate-authority-data"

validate.sh

#!/bin/bash

set -xeuo pipefail

exit 0

gitlab.ci.yml

variables:

HTTPS\_PROXY: http://securecomms.int.dh:3128

HTTP\_PROXY: http://securecomms.int.dh:3128

ORDER\_ID:

description: "The Digicert order id we want to renew"

STACK:

description: "`<stack\_name>/config` file from Cloud > clients > client E.g., `cds-wfm-pre` for internal.pre.cds-wholefoods"

default:

image: docker.artifactory.dunnhumby.com/cs-core/ceng-deploy:master

tags:

- gke-shared-kubernetes

stages:

- renew

- rotate

- validate

- revoke

renew-cert:

stage: renew

rules:

- when: manual

script:

- source renew\_cert.sh $ORDER\_ID

- echo "RENEWED\_ORDER\_ID=$RENEWED\_ORDER\_ID" >> cert.env

- echo "EXPIRES\_AT=$EXPIRES\_AT" >> cert.env

artifacts:

reports:

dotenv: cert.env

publish-new-cert-pem:

stage: renew

script:

- ./download\_pem.sh $RENEWED\_ORDER\_ID > cert.pem

- cp cert.pem cert.cer

- cp cert.pem cert.crt

needs: ["renew-cert"]

artifacts:

paths:

- cert.cer

- cert.crt

- cert.pem

rotate-istio-tls-secret:

stage: rotate

script:

- echo $GCP\_APP\_CREDS\_JSON > ~/gcp-app-creds-key.json

- GOOGLE\_APPLICATION\_CREDENTIALS=~/gcp-app-creds-key.json

- GOOGLE\_APPLICATION\_CREDENTIALS\_CONTENT=${GCP\_APP\_CREDS\_JSON}

- gcloud auth activate-service-account --key-file=$GOOGLE\_APPLICATION\_CREDENTIALS

- source get\_core\_config.sh ${STACK}

- gcloud config set project ${PROJECT}

- gsutil cp gs://gcp-project-certs/internal.${WEBUSERPROXY\_DOMAIN}/internal.${WEBUSERPROXY\_DOMAIN}.key cert.key

- bash setup\_kubeconfig.sh --region=${REGION} --cluster\_name=${CLUSTER\_NAME} --project=${PROJECT} --master\_custom\_dns=${CLUSTER\_DNS}

- kubectl delete secret -n istio-system istio-ingressgateway-certs

- kubectl create secret -n istio-system tls istio-ingressgateway-certs --key cert.key --cert cert.crt

- kubectl delete pod -n istio-system -l app=istio-ingressgateway -l istio=ingressgateway

needs: ["publish-new-cert-pem"]

validate:

stage: validate

script:

- ./validate.sh

needs: ["renew-cert", "rotate-istio-tls-secret"]

revoke:

stage: revoke

script:

- ./revoke.sh

needs: ["validate"]