

**cors.xml(GraphQL Pod )**

vi configDropins/overrides/cors.xml

<server>

<!-- https://www.ibm.com/support/knowledgecenter/en/SSEQTP\_liberty/com.ibm.websphere.liberty.autogen.nd.doc/ae/rwlp\_config\_cors.html -->

<cors domain="/content-services-graphql"

allowedOrigins="Salesforce Domain"

allowedMethods="GET, POST, OPTIONS"

allowedHeaders="Connection,Pragma,Cache-Control,ECM-CS-XSRF-Token,XSRFtoken,Origin,User-Agent,Content-Type,Content-Length,Accept-Control-Request-Method,Accept-Control-Request-Headers,Accept,Referer,Accept-Encoding,Accept-Language,DNT,Host,Content-Length,Cache-control,Cookie,Authorization,Access-Control-Allow-Origin"

exposeHeaders="Authorization,Content-Disposition,Content-Length,Content\_Type,Content-Language,X-Powered-By,Date,Allow,Transfer-Encoding,$WSEP,DNT,Access-Control-Allow-Credentials,Access-Control-Allow-Headers,Access-Control-Allow-Max-Age,Access-Control-Allow-Methods,Access-Control-Allow-Origin,Access-Control-Expose-Headers,Connection,Cache-control,Cookie"

allowCredentials="true"

maxAge="3600" />

</server>

**ibm-nginx Pod**

vi /user-home/\_global\_/nginx-conf.d/cp4ba-graphql-zen-extension.conf

location /content-services-graphql

{

if ($request\_method = 'OPTIONS') {

add\_header 'Access-Control-Allow-Origin' $http\_origin;

add\_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS';

#

# Custom headers and headers various browsers \*should\* be OK with but aren't

#

add\_header 'Access-Control-Allow-Headers' 'Authorization,ecm-cs-xsrf-token,X-ECM-SF-ORG-ID,DNT,User-Agent,X-Requested-With,If-Modified-Since,Cache-Control,Content-Type,Range';

#

# Tell client that this pre-flight info is valid for 20 days

#

add\_header 'Access-Control-Max-Age' 1728000;

add\_header 'Access-Control-Allow-Credentials' 'true';

add\_header 'Content-Type' 'text/plain; charset=utf-8';

add\_header 'Content-Length' 0;

return 204;

}

if ($request\_method = 'GET'){

add\_header 'Access-Control-Allow-Origin' $http\_origin;

}

if ($request\_method = 'POST'){

add\_header 'Access-Control-Allow-Origin' $http\_origin;

}

access\_by\_lua\_block {

local jwtcheck = require("nginx\_data.verify-jwt")

local auth\_header = ngx.var.http\_Authorization

if (auth\_header == nil) then

jwtcheck.auth()

elseif (string.sub(auth\_header,1,5)~="Basic") then

jwtcheck.auth()

end

}

proxy\_set\_header Host $http\_host;

proxy\_set\_header X-ZEN-ROUTING '$host';

proxy\_set\_header Access-Control-Allow-Origin '$http\_origin';

proxy\_pass https://cp4ba-graphql-svc.cp4ba.svc:9443/content-services-graphql;

proxy\_buffer\_size 256k;

proxy\_connect\_timeout 300;

proxy\_read\_timeout 300;

proxy\_send\_timeout 300;

proxy\_buffers 8 512k;

proxy\_busy\_buffers\_size 512k;

}

location /content-services-graphql/ping

{

if ($request\_method = 'OPTIONS') {

add\_header 'Access-Control-Allow-Origin' $http\_origin;

add\_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS';

#

# Custom headers and headers various browsers \*should\* be OK with but aren't

#

add\_header 'Access-Control-Allow-Headers' 'Authorization,ecm-cs-xsrf-token,X-ECM-SF-ORG-ID,DNT,User-Agent,X-Requested-With,If-Modified-Since,Cache-Control,Content-Type,Range';

#

# Tell client that this pre-flight info is valid for 20 days

#

add\_header 'Access-Control-Max-Age' 1728000;

add\_header 'Access-Control-Allow-Credentials' 'true';

add\_header 'Content-Type' 'text/plain; charset=utf-8';

add\_header 'Content-Length' 0;

return 204;

}

if ($request\_method = 'GET'){

add\_header 'Access-Control-Allow-Origin' $http\_origin;

}

if ($request\_method = 'POST'){

add\_header 'Access-Control-Allow-Origin' $http\_origin;

}

proxy\_set\_header Host $http\_host;

proxy\_set\_header Access-Control-Allow-Origin '$http\_origin';

proxy\_set\_header X-ZEN-ROUTING '$host';

proxy\_pass https://cp4ba-graphql-svc.cp4ba.svc:9443/content-services-graphql/ping;

proxy\_buffer\_size 256k;

proxy\_connect\_timeout 300;

proxy\_read\_timeout 300;

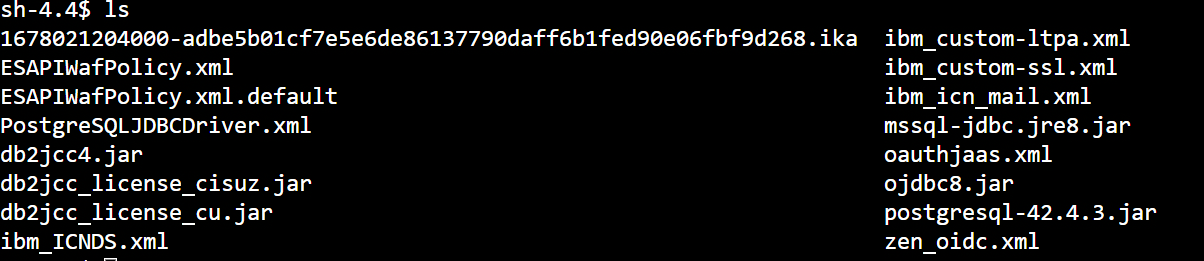
proxy\_send\_timeout 300;

proxy\_buffers 8 512k;

proxy\_busy\_buffers\_size 512k;

}

vi configDropins/overrides/ESAPWafPolicy.xml



<?xml version="1.0" encoding="UTF-8"?>

<policy>

<aliases/>

<settings>

<mode>block</mode>

<error-handling>

<default-redirect-page>/error.jsp</default-redirect-page>

<block-status>500</block-status>

</error-handling>

</settings>

<virtual-patches>

<!-- Uncomment and update to add return host validation when enableOAuthProxy is set to true in web.xml

<virtual-patch id="oauth2-return" path=".\*/jaxrs/oauth2/.\*"

variable="request.parameters.state"

pattern="^(localhost|localhost:http)$"

message="Detected invalid OAuth2 return host.">

</virtual-patch>

-->

</virtual-patches>

<outbound-rules>

<add-header name="Cache-Control" value="no-cache, no-store" path=".\*/|.\*\.jsp|.\*/jaxrs/.\*|.\*/api/.\*"/>

<add-header name="Content-Security-Policy" value="default-src 'self' blob: https:; font-src 'self' data: blob: https:; img-src 'self' data: blob: https:; script-src 'self' 'unsafe-inline' 'unsafe-eval' https:; worker-src 'self' blob: https:; style-src 'self' 'unsafe-inline' https:; frame-ancestors 'self'; object-src 'self' https:https://standardbankgroupltd--chnwmcdev.sandbox.lightning.force.com:\*" path="/.\*"/>

<add-header name="Referrer-Policy" value="same-origin" path="/.\*"/>

<add-header name="Strict-Transport-Security" value="max-age=7776000; includeSubdomains" path="/.\*"/>

<add-header name="X-Content-Type-Options" value="nosniff" path="/.\*"/>

<!-- <add-header name="X-Frame-Options" value="sameorigin" path="/.\*"/> -->

<add-header name="X-Permitted-Cross-Domain-Policies" value="none" path="/.\*"/>

<add-header name="X-XSS-Protection" value="0" path="/.\*"/>

<add-http-only-flag>

<cookie name="ECM-XSRF-Token"/>

</add-http-only-flag>

<add-secure-flag>

<cookie name="ECM-XSRF-Token"/>

</add-secure-flag>

</outbound-rules>

<url-rules>

<!-- Deny HTTP methods other than GET, POST, PUT, DELETE, HEAD, and OPTIONS, and only allow POST for logon requests -->

<restrict-method deny="^((?!GET|POST|PUT|DELETE|HEAD|OPTIONS).)\*$"/>

<restrict-method deny="^(GET|PUT|DELETE|HEAD|OPTIONS)$" path=".\*/jaxrs(/|/.+/)logon$"/>

</url-rules>

</policy>

**Find keystore password**

1. Ibm-fncm-secret
2. Download trust store

Secret 🡪cp4ba-fncm-custom-ssl-secret 🡪 download ibm\_customFNCMTrustStore.p12

<https://www.ibm.com/support/pages/how-convert-p12-keystore-jks-keystore>

cd /mnt/c/users/ZZ01LA864/Desktop

keytool -v -list -keystore ibm\_customFNCMKeyStore.p12| grep Alias -C 3

keytool -importkeystore -srckeystore ibm\_customFNCMKeyStore.p12 -srcstoretype PKCS12 -destkeystore ibm\_customFNCMTrustStore.jks -deststoretype JKS -srcstorepass Passw0rd -deststorepass Passw0rd -srcalias 1 -destalias jkscert -srckeypass Passw0rd -destkeypass Passw0rd -noprompt

oc login --token=sha256~I\_7Yhh8qIwkvlh4cQhnW8Xusv8wAoOvcoP3ZAg0CxPI --server=https://c100-e.eu-gb.containers.cloud.ibm.com:31137

1. <https://github.ibm.com/dte-support/public/blob/master/CloudPaks/cp4i-certs.md>
2. oc extract secret/$(oc get route console -n openshift-console -o jsonpath="{.spec.host}" | awk -F'.' '{print $2}') -n ibm-cert-store --confirm --to=./
3. For Ubuntu  
   csplit -sf cert -n 1 tls.crt '/-----BEGIN CERTIFICATE-----/' {\*}
4. oc -n ibm-common-services delete certificate route-cert
5. oc -n ibm-common-services delete secret route-tls-secret
6. oc -n ibm-common-services create secret generic route-tls-secret --from-file=ca.crt=./cert2 --from-file=tls.crt=./cert1 --from-file=tls.key=./tls.key
7. oc patch AutomationUIConfig iaf-system --type merge --patch '{"spec":{"tls":{"issuerRef":{"name":""}}}}'
8. oc delete certificate iaf-system-automationui-aui-zen-ca
9. oc delete certificate iaf-system-automationui-aui-zen-cert
10. oc delete secret external-tls-secret
11. oc create secret generic external-tls-secret --from-file=cert.crt=./tls.crt --from-file=cert.key=./tls.key
12. oc -n ibm-common-services delete pod -l app=auth-idp
13. oc delete pod -l component=ibm-nginx
14. oc delete route cpd
15. oc create route passthrough cpd --service=ibm-nginx-svc --port=ibm-nginx-https-port