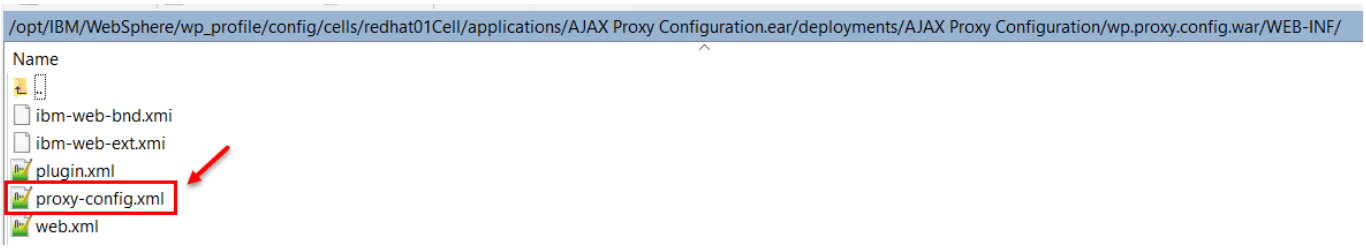
En este post vamos a ver como configurar un servicio rest en el Ajax Proxy de Portal 8.5

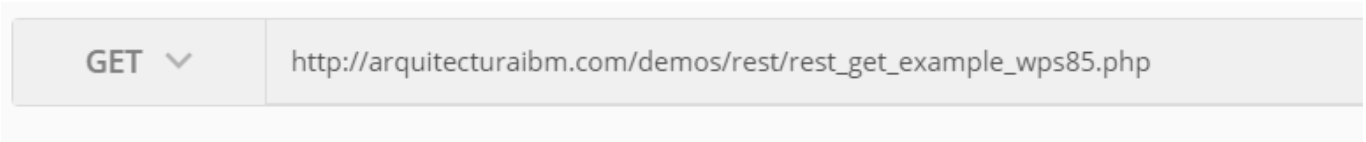
1. Descargar el archivo [***proxy-config.xml***]
2. Configurar nuestro servicio en el archivo [**proxy-config.xml**]
3. Subir el archivo [***proxy-config.xml***] a una carpeta propia dentro de Portal 8.5
   Example: [***/opt/IBM/WebSphere/wp_profile/ibmperu/***] y el archivo quedaría aquí:
   [*/opt/IBM/WebSphere/wp_profile/ibmperu/proxy-config.xml*]
4. Ejecutar el **ConfigEngine.sh(Linux)** o **ConfigEngine.bat(Linux)**
5. Reiniciar Portal

_1. En **IBM WebSphere Portal 8.5**, ingresar vía **WinSCP** a la ruta y descargar el archivo [/opt/IBM/WebSphere/wp_profile/config/cells/**redhat01Cell**/applications/AJAX Proxy Configuration.ear/deployments/AJAX Proxy Configuration/wp.proxy.config.war/WEB-INF/**proxy-config.xml**]



_2. Configuramos el servicio rest en

Yo dispongo de un servicio rest: [https://arquitecturaibm.com/demos/rest/rest_get_example_wps85.php] y al acceder por GET devuelve lo siguiente

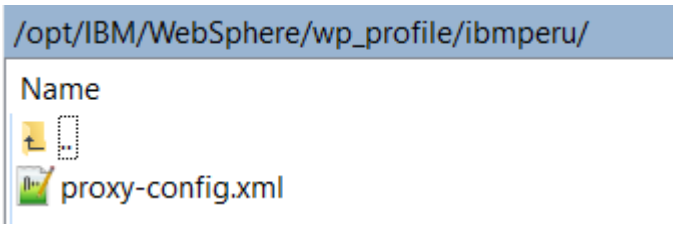



```
{
    "wps_username": "jupari-2018",
    "wps_password": "limaperu-2018"
}
```

Editar el archivo [***proxy-config.xml***] y solo agregamos lo resaltado en amarillo, debería quedar así. Igual adjunto el xml (proxy-config-custom.zip)

```
<?xml version="1.0" encoding="UTF-8"?>
<proxy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http://www.ibm.com/

    <mapping contextpath="/proxy" url="*" name="proxy"/>
    <mapping contextpath="/myproxy" url="*" name="myproxy"/>
    <mapping contextpath="/common_proxy" url="*" name="common"/>
    <mapping contextpath="/cmis_proxy" url="*" name="cmis" />

    <policy url="http://www.ibm.com/*" name="ibm1">
        <actions>
            <method>GET</method>
        </actions>
    </policy>
    <policy url="http://www-03.ibm.com/*" name="ibm2">
        <actions>
            <method>GET</method>
        </actions>
    </policy>
    <policy url="http://www.redbooks.ibm.com/*" name="redbooks">
        <actions>
            <method>GET</method>
        </actions>
    </policy>

    <policy url="http://arquitecturaibm.com/*" name="arquitecturaibm_rest_get">
        <actions>
            <method>GET</method>
        </actions>
    </policy>

    <policy url="{$default_ltpa_policy}" name="default_ltpa">
        <actions>
            <method>GET</method>
            <method>HEAD</method>
        </actions>
        <cookie-rule name="ltpa">
            <cookie>{$token.ltpa.name}</cookie>
            <cookie>{$token.ltpa2.name}</cookie>
        </cookie-rule>
    </policy>
    <policy url="{$default_policy}" name="default">
        <actions>
            <method>GET</method>
            <method>HEAD</method>
        </actions>
    </policy>
```

_3. Subir el archivo a [/opt/IBM/WebSphere/wp_profile/ibmperu] y debería quedar así



_4. Ahora ejecutar el comando

cd /wp_profile/ConfigEngine/

./ConfigEngine.sh -DWasPassword=wpsadmin -DPortalAdminPwd=wpsadmin checkin-wp-

proxy-config -DProxyConfigFileName=/opt/IBM/WebSphere/wp_profile/ibmperu/proxy-config.xml

_5. Reiniciar Portal

_6. Probando

Usuarios anonimos
http://192.168.227.144:10039/**wps**/**proxy**/**http**/**arquitecturaibm.com/demos/rest/rest_get_example_wps85.php**

Usuarios autenticados
http://192.168.227.144:10039/**wps**/**myproxy**/**http**/**arquitecturaibm.com/demos/rest/rest_get_example_wps85.php**

Escenario 1:

Tenemos un Script Portlet que quiere consumir directamente el servicio rest apuntando a [https://arquitecturaibm.com/demos/rest/rest_get_example_wps85.php], al querer consumir le saldrá un error de cross-origin

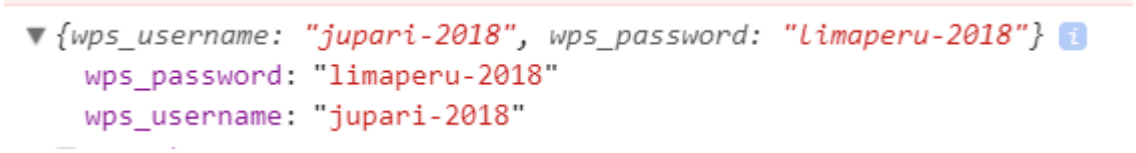
```
$.ajax({
    url: "http://arquitecturaibm.com/demos/rest/rest_get_example_wps85.php",
    type: 'GET',
    success: function(res) {
        console.log(res);
    }
});
```

```
Failed to load http://arquitecturaibm.com/demos/rest/rest_get_example_wps85.php: No 'Access-Control-Allow-Origin' header is present on the requested resource. Origin 'http://192.168.227.144:10039' is therefore not allowed access.
>
```

La primera alternativa de solución sería poner en el header del servicio rest lo siguiente [header('Access-Control-Allow-Origin: *');] y con eso ya no tendríamos temas de cross-origin. Pero lamentablemente cuando no tenemos control del servicio rest no podremos hacer nada.

La mejor alternativa es configurar el servicio en el proxy de Portal como lo hemos hecho y tendríamos este codigo

```
$.ajax({
    url: "http://192.168.227.144:10039/wps/myproxy/http/arquitecturaibm.com/demos/rest/rest_get_example_wps85.php",
    type: 'GET',
    success: function(res) {
        console.log(res);
    }
});
```



¿Qué más con el Ajax Proxy de Portal 8.5?

1. Podemos configurar http-basic vía credential-vault de Portal
2. Podemos configurar HTTPS
3. Podemos configurar Cookies
4. Podemos configurar GET, POST, HEAD, DELETE, PUT
5. Podemos configurar meta-data
6. Podemos configurar el mime-type de comunicación

Fuentes

- https://www.ibm.com/support/knowledgecenter/en/SS6KJL_8.5.0/FEB/in_portlet_before_you_install_feb_portlet.html
- https://portal2portal.blogspot.pe/2013/01/testing-ajax-proxy-in-websphere.html
- http://portalkrishna.blogspot.pe/2014/10/configuring-ajax-proxy-in-websphere.html
- https://www.youtube.com/watch?v=3vUjvVZupxs

Julio Pari (IT Architect IBM)

Si te ha interesado este artículo y deseas un apoyo o asesoría en algún requerimiento, envíame un mensaje a: (info@juliopari.com) o sino a través de Linkedin: https://www.linkedin.com/in/juliopari/
