



ZCADMIN – IBM z/OS Connect Administration

WebSphere Liberty Profile with
IBM z/OS Connect (OpenAPI 2) and/or
IBM z/OS Connect (OpenAPI 3)
Administration

Mitch Johnson
mitchj@us.ibm.com
Washington Systems Center

mitchj@us.ibm.com



© 2017, 2023 IBM Corporation
Slide 1

Notes and Disclaimers

- The information in this presentation was derived from various product documentation web sites.
- Additional information included in this presentation was distilled from years of experience implementing security using RACF with z/OS products like CICS, IMS, Db2, MQ, etc. as well as Java runtimes environments like WebSphere Application Server and WebSphere Application Server Liberty which is commonly called Liberty.
- There will be additional information on slides that will be designated as Tech/Tips. These contain information that at perhaps at least interesting and hopefully, useful to the reader.
- **IBM z/OS Connect** (**OpenAPI 2**) refers to the z/OS Connect EE product prior to service level V3.0.55. **IBM z/OS Connect** (**OpenAPI 3**) refers to the additional functions and features added with service level V3.0.55. Important - servers configured for OpenAPI 2 can will continue to operate as is with service level V3.0.55 and later.
- A z/OS  or a Java  or a Liberty  or a z/OS Connect OpenAPI 2,  or a z/OS Connect OpenAPI 3  icon will appear on slides where the information is specific to these products. Don't hesitate to ask questions as to why the icon does or does not appear on certain slides.
- The examples, tips, etc. present in this material are based on firsthand experiences and are not necessarily sanctioned by Liberty or z/OS Connect development.

Agenda

- **OMVS, Liberty, z/OS Connect configuration**
- **RACF, Liberty and z/OS Connect Security**
- **Connecting z/OS Connect servers to other z/OS subsystems**
- **Useful Liberty features and MVS commands**
- **Where do I look when things go wrong?**
- **Managing and Monitoring Liberty and z/OS Connect**
- **Additional Material - sample administrative JCL**

**Let's start by reviewing some of the basic Liberty,
OMVS, z/OS Connect configuration details and options**

Begin by verifying the Java and OMVS environments are ready*

Basic system configuration settings which have more than once caused issues

- Prevent out-of-memory or other storage issues:
 - Verify the Java runtime is not being limited by system parameters, e.g., *MAXASSIZE* (2 147 483 647), *MAXTHREADS*, etc., for details see *BPXPRM setting* at URL https://www.ibm.com/docs/en/sdk-java-technology/8?topic=SSYKE2_8.0.0/com.ibm.java.vm.80.doc/docs/j9_configure_zos_bpxprm.html
 - Check the value of *ASSIZEMAX* in the OMVS segments of the identities involved and ensure it is adequate, see *MAXASSIZE* above.
 - Exclude OMVS from any IEFUSI exit, SUBSYS(OMVS,NOEXITS) in PARMLIB member *SMFRPMxx*.
 - Verify the JCL *MEMLIMIT* parameter is within reason for your system.
- Start an OMVS shell session and verify that Java is fully operational by entering command *java -version*, you see should results like this:

```
java version "1.8.0_301"
Java(TM) SE Runtime Environment (build 8.0.6.35 - pmz6480sr6fp35-20210714_01(SR6 FP35))
IBM J9 VM (build 2.9, JRE 1.8.0 z/OS s390x-64-Bit Compressed References 20210622_7763 (JIT enabled, AOT
enabled)
OpenJ9   - b1f3adb
OMR      - c2f4a18
IBM      - c24a144
JCL - 20210625_01 based on Oracle jdk8u301-b09
```

- Verify that RACF identities associated with started tasks have OMVS segments with UIDs and GIDs and valid HOME directories and that the identities can invoke Java commands.
- Verify the *zconsetup* script has been executed. My recommendation is to execute this script in the SMP/E target environment, otherwise it will be lost when service is applied and propagated to other images.

*be sure the power plug is plugged into the outlet

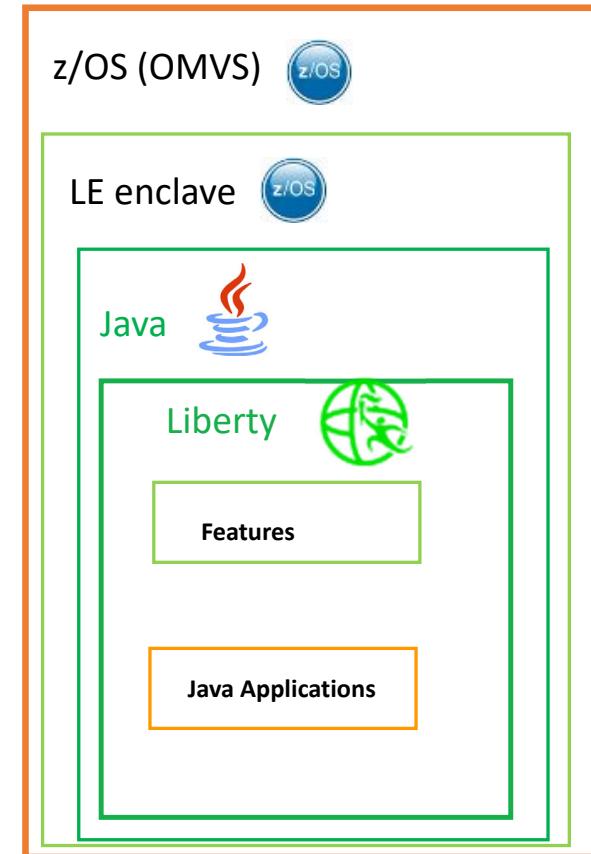


Think of a Liberty server consisting of layers of software products

- A Liberty server contains
 - Java applications
 - Features which provide shared access to basic z/OS functions, e.g., SAF security, WLM, RRS, SMF etc., for multiple Java applications running concurrently.
- Started by an OMVS script that starts the Java environment (as an OMVS process).
- Runs in a Language Environment (LE) enclave configured to support OMVS and Java processes.
- On z/OS, provides access to z/OS services and facilities (e.g., SAF, WLM, RRS, SMF, JCL, started tasks, etc.)

Knowing the different layers and their relationship is important regarding

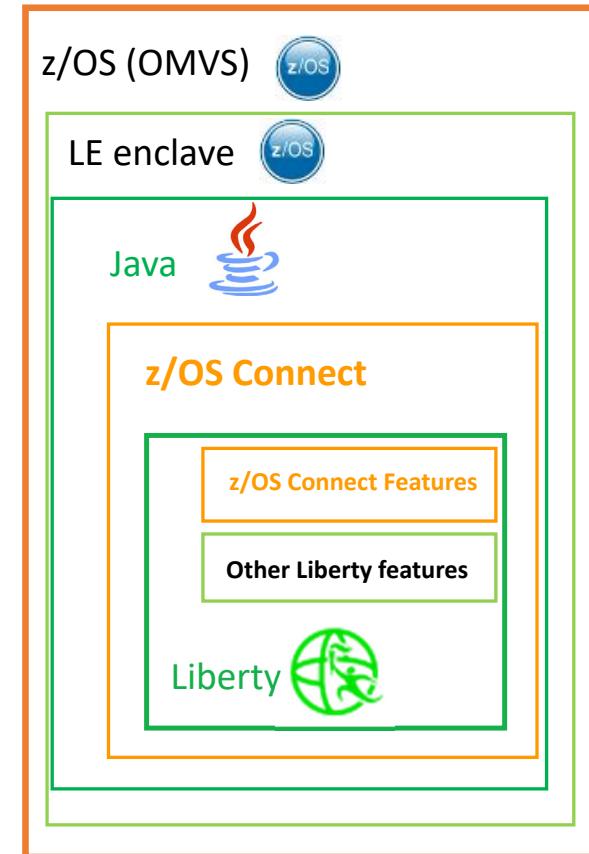
- Understanding which layer a configuration options, e.g., environment variables, etc., applies.
- Monitoring and understanding the health of the server
- Performing problem determination and performance tuning





A z/OS Connect Liberty server adds an additional layer

- **z/OS Connect also provides Java code that initiates a Liberty process***
- z/OS Connect is also provides Liberty features
- The z/OS Connect Liberty features provide client access to common z/OS subsystems, e.g., CICS, Db2, IMS, MQ, etc.



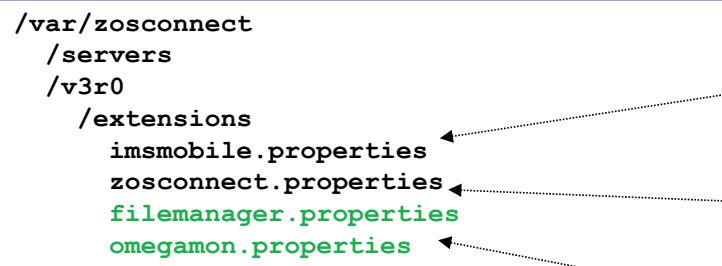
* z/OS Connect starts a Liberty process using a system programming interface (SPI). See the Note regarding environment variables and jvm.options and server.env files at URL <https://www.ibm.com/docs/en/was-liberty/zos?topic=liberty-embedding-server-in-your-applications> regarding restrictions in this environment.



The `zconsetup` script is invoked once per LPAR per z/OS Connect product path

The `zconsetup` script creates a symbolic link from the WLP `..v3r0/wlp/etc` directory in the product path (normally R/O) to a local R/W directory (creating a default configuration and local extension directories). This provides a means for administrator to configure the image.

```
JOHNSON:/usr/lpp/IBM/zosconnect/v3r0/wlp/etc: ls -al
total 32
drwxrwxr-x  2 OMVSKERN 0          8192 Jun 24 10:24 .
drwxrwxr-x 10 OMVSKERN 0          8192 Jun 24 10:24 ..
lrwxrwxrwx  1 990023  0          31 Jul 27 2020 extensions -> /var/zosconnect/v3r0/extensions
```



com.ibm.websphere.productId=com.ibm.ims.mobile
com.ibm.websphere.productInstall=imsmobile/wlp-ext

com.ibm.websphere.productId=com.ibm.zosconnect
com.ibm.websphere.productInstall=runtime

**Relative v.
absolute path**

Properties and options for services providers, exits, etc. not shipped or
embedded in the z/OS Connect directory structure.

com.ibm.websphere.productId=omegamonRequestMonitor-2.0
com.ibm.websphere.productInstall=/var/zosconnect/omegamon

- This directory structure and contents is created by invoking the `zconsetup` script and **must be created on each LPAR** on which z/OS Connect will execute. This is how the z/OS Connect Liberty server locates service provider executables. Note: the `com.ibm.websphere.productInstall` directive value that is relative to directory `/usr/lpp/IBM/zosconnect/v3r0`.
- Not creating this link will cause messages `CWWKF0001E: A feature definition could not be found for zosconnect:....` or `CWWKE0054E: Unable to open /usr/lpp/IBM/zosconnect/v3r0/wlp/etc/extensions/zosconnect.properties`



A Liberty server is created by using the *server* command

To create a new Liberty server, use the *server create* command, as in:

server create serverName

- Where *serverName* is any value you wish, such as *wlpopsrv* or *wlpOpenIDAuthServer* and this value will be the name of the server instance. The default value is *defaultServer*
- Environment variable *WLP_USER_DIR* must be set to determine the location of the configuration directory and files created by this command. The constant *servers* is appended to the value of this variable, e.g., *{\$WLP_USER_DIR}/servers* and the server's name is appended to this root directory and full directory path is the location where the server's configuration files, and default directories are created, e.g., *{\$WLP_USER_DIR}/servers/serverName*. The *WLP_USER_NAME* variable is required when starting a server and must be the same value used when the server was created. There is no default value for a Liberty server.

Note: the name of the server does not have to be same as the started task name, as shown in this example (note how the value for *WLP_USER_DIR* is provided by the *PATH* of the *WLPUDIR* DD statement):

```
//WLPOPID PROC PARM='wlpOpenIDAuthServer'  
//  
// SET INSTDIR='/usr/lpp/liberty_zos/18.0.0.1'  
// SET USERDIR='/var/wlp'  
//  
//STEP1 EXEC PGM=BPXBATSL,REGION=0M,TIME=NOLIMIT,  
// PARM='PGM &INSTDIR./lib/native/zos/s390x/bbgzsrv &PARMS'  
//WLPUDIR DD PATH='&USERDIR.'  
//STDOUT DD SYSOUT=*  
//STDERR DD SYSOUT=*  
//MSGLOG DD SYSOUT=*  
//STDENV DD PATH='/etc/system.env',PATHOPTS=(ORDONLY)
```



Creating a server creates the initial server configuration file, e.g., server.xml

A sample server.xml configuration file

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="simple server">

    <!-- Enable features -->
    <featureManager>
        <feature>apiDiscovery-1.0</feature>
        <feature>appSecurity-2.0</feature>
        <feature>beanValidation-1.0</feature>
        <feature>jaxb-2.2</feature>
        <feature>jaxrs-1.1</feature>
        <feature>jsf-2.0</feature>
        <feature>jsp-2.3</feature>
    </featureManager>

    <httpEndpoint id="defaultHttpEndpoint"
                  httpPort="29080"
                  host="*"
                  httpsPort="29443" />

    <!-- Automatically expand WAR files and EAR files -->
    <applicationManager autoExpand="true"/>

    <enterpriseApplication
        id="Trader_EAR"
        location="Trader_EAR.ear"
        name="Trader_EAR">
        <classloader delegation="parentLast"/>
    </enterpriseApplication>

</server>
```

Add or remove features as needed in the featureManager configuration elements.

Configure connectivity in httpEndpoint configuration elements

Add other configuration elements as needed.



A z/OS Connect Liberty is created by using the `zosconnect` command

To create a z/OS Connect server, use the `zosconnect` command using one of these templates, as in:

```
zosconnect create serverName --template=templateName
```

Where `templateName` can be:

- `zosconnect:apiRequester` for an OpenAPI2 z/OS Connect API requester enabled server
- `zosconnect:default` template for base OpenAPI2 z/OS Connect servers
- `zosconnect:openApi3` template for base OpenAPI3 z/OS Connect native servers
- `zosconnect:openApi3Requester` template for base OpenAPI3 z/OS Connect native API requester servers

- Where `serverName` is any value you wish, such as `zceesrvr` or `zCEEServer`, and this value will be the name of the server instance. The templates can be found in directory `/usr/lpp/IBM/zosconnect/v3r0/runtime/templates/servers`.
- Environment variable `WLP_USER_DIR` will be used to set the location of the configuration directory and files created by this command, default location is `/var/zosconnect/servers` where `/var/zosconnect` is default value for `WLP_USER_DIR` for z/OS Connect.
- The `zosconnect:openApi3` template installs feature `zosconnect:zosConnect-3.0`. z/OS Connect service provider features, e.g., `zosconnect:cics-1.0`, `zosconnect:mqService-1.0`, `zosconnect:dbService-1.0` and `imsmobile:imsmobile-2.0` have dependencies on feature `zosconnect:zosConnect-2.0` and are not compatible with feature `zosconnect:zosConnect-3.0`. z/OS Connect XML configuration attributes other than `zosconnect_cicsIpicConnection` and `zosconnect_db2Connection` are not recognized in an z/OS Connect OpenAPI 3 server.

There are other templates, but they are essentially only useful as samples of service provider configuration options.



What is the significance of the OpenAPI Specification 2.0 and 3.0 for z/OS Connect?

The industry standard framework for describing REST APIs

The OpenAPI Initiative (OAI) was created by a consortium of forward-looking industry experts who recognize the immense value of standardizing on how APIs are described. As an open governance structure under the Linux Foundation, the OAI is focused on creating, evolving and promoting a vendor neutral description format. The OpenAPI Specification was originally based on the [Swagger Specification](#), donated by SmartBear Software.

- **z/OS Connect and Swagger 2.0 (Open API Specification 2), supported initially by z/OS Connect**

Initially, accessing z/OS resources was the only desire for developing APIs. The interactions with the z/OS resources was driven by the layout of the CICS COMMAREA or CONTAINER, the IMS or MQ messages or the Db2 REST service.

- The details of the interactions with the z/OS resource determined the contents of the API request and response messages and the subsequent specification document.
- **z/OS Connect produces the specification document that describes the methods and request and response messages.**

- **z/OS Connect and Open API Specification 3, supported by z/OS Connect starting in March 2022 service, V3.0.55**

As companies mature their API strategy, they begin to introduce API governance boards to drive consistency in their API design. As more public APIs are created, government and industry standards bodies begin to regulate and drive for standardization. This drives the need for “API first” functional mapping capabilities within the integration platform. The external API design determined the layouts of the API request and response messages provided by the specification documents which was consumed by z/OS Connect to describe the z/OS resource interactions.

- The API details of the methods and layouts of request and response messages are provided in advance and access to the z/OS resource is driven by the API design
- **z/OS Connect consumes the specification document that describes the methods and request and response messages**



Contrasting the OpenAPI 2 /OpenAPI 3 specification

z/OS Connect produces an OpenAPI 2 specification document, which is driven by the nature of the z/OS asset (JSON Format)

The image shows two side-by-side Notepad windows. The left window is titled "cscvinc.json - Notepad" and contains the JSON representation of the OpenAPI 2 specification. The right window is titled "cscvinc.yaml - Notepad" and contains the YAML representation of the OpenAPI 3 specification. Both windows show the same API definition for "cscvinc". Red circles highlight specific sections in both files: one circle covers the "tags" section under the "post" operation in the YAML file, and another circle covers the "tags" section under the "get" operation in the JSON file.

```

cscvinc.json - Notepad
{
  "swagger": "2.0",
  "info": {
    "description": "",
    "version": "1.0.0",
    "title": "cscvincapi"
  },
  "basePath": "/cscvincapi",
  "schemes": [
    "https",
    "http"
  ],
  "consumes": [
    "application/json"
  ],
  "produces": [
    "application/json"
  ],
  "paths": {
    "/employee/{employee}": {
      "get": {
        "tags": [
          "cscvincapi"
        ],
        "operationId": "getCscvincSelectService",
        "parameters": [
          {
            "name": "Authorization",
            "in": "header",
            "required": false,
            "type": "string"
          },
          {
            "name": "employee",
            "in": "path",
            "required": true,
            "type": "string",
            "maxLength": 6
          }
        ],
        "responses": {
          "200": {
            "description": "OK",
            "schema": {
              "$ref": "#/definitions/getCscvincSelectService_response_200"
            }
          },
          "404": {
            "description": "Not Found",
          }
        }
      }
    }
  }
}

cscvinc.yaml - Notepad
openapi: 3.0.1
info:
  title: cscvinc
  description: ""
  version: 1.0.0
servers:
- url: /cscvinc
x-ibm-zcon-roles-allowed:
- Manager
paths:
  /employee:
    post:
      tags:
        - cscvinc
      operationId: postCscvincInsertService
      x-ibm-zcon-roles-allowed:
        - Staff
      parameters:
        - name: Authorization
          in: header
          schema:
            type: string
      requestBody:
        description: request body
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/postCscvincInsertService_request'
            required: true
      responses:
        200:
          description: OK
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/postCscvincInsertService_response_200'
              x-codegen-request-body-name: postCscvincInsertService_request
  /employee/{employee}:
    get:
      tags:
        - cscvinc
      operationId: getCscvincSelectService
      x-ibm-zcon-roles-allowed:
        - Staff
      parameters:
        - name: Authorization
          in: header
          schema:
            type: string

```

z/OS Connect consumes an OpenAPI specification document, driven by the design of the API (YAML Format*)



There are differences between the initial server.xml configuration files

```
default template - OpenAPI 2 server.xml configuration file
<?xml version="1.0" encoding="UTF-8"?>
<server description="new server">
    <!-- Enable features -->
    <featureManager>
        <feature>zosconnect:zosConnect-2.0</feature>
        <feature>zosconnect:zosConnectCommands-1.0</feature>
        <feature>apiDiscovery-1.0</feature> *
    </featureManager>

    <!-- To access this server from a remote client add a host attribute
    <httpEndpoint id="defaultHttpEndpoint"
        host="*"
        httpPort="9080"
        httpsPort="9443" />
    <!-- add cors to allow cross origin access, e.g. when using swagger UI
    to fetch swagger doc from zOS Connect Enterprise Edition -->
    <cors id="defaultCORSConfig"
- - - - - 24 Line(s) not Displayed

    <!-- config requires updateTrigger="mbean" for REFRESH command support
-->
<config updateTrigger="mbean" monitorInterval="500"/>

    <zosconnect_zosConnectManager setUTF8ResponseEncoding="true"/>

    <!-- zosConnect APIs -->
    <zosconnect_zosConnectAPIs updateTrigger="disabled" pollingRate="5s"
        <!-- zosConnect Services -->
    <zosconnect_services updateTrigger="disabled" pollingRate="5s"/>

    <!-- applicationMonitor is not applicable for z/OS Connect EE servers --
-->
    <applicationMonitor updateTrigger="disabled" dropinsEnabled="false"/>

</server>
```

* Include these features if not already present.

```
openApi3 template - OpenAPI 3 server.xml configuration file
<?xml version="1.0" encoding="UTF-8"?>
<server description="new server">
    <!-- Enable features -->
    <featureManager>
        <feature>zosconnect:zosConnect-3.0</feature>
        <feature>openapi-3.0</feature>
    </featureManager>

    <!-- To access this server from a remote client add a host attribute
    <httpEndpoint id="defaultHttpEndpoint"
        host="*"
        httpPort="9080"
        httpsPort="9443" />
    - - - - - 12 Line(s) not Displayed
        <!-- config requires updateTrigger="mbean" for REFRESH command support
        config updateTrigger="mbean"/>

        <!-- applicationMonitor requires updateTrigger="mbean" for REFRESH command
        support -->
        <applicationMonitor updateTrigger="mbean" dropinsEnabled="false"/>

        <!-- Automatic expansion of WAR files is required for z/OS Connect native
        servers running the zosConnect-3.0 feature -->
        <applicationManager autoExpand="true" />

        <!-- APIs are deployed as WAR files and a webApplication element must be
        used to specify the location of the API WAR and optionally the name of the API
        -->
        <webApplication id="My API" location="${server.config.dir}/apps/api.war"
            name="MyAPI"/>

    </server>
```

Note there are no *zosconnect* or *cors* configuration elements present with Open API 3.

Tech Tip: Use multiple mount points and dedicated ZFS file systems

Create the mount points and mount file systems prior to running the zconsetup script

```
mkdir -p /var/zosconnect
mkdir -p /var/zosconnect/servers
mkdir -p /var/zosconnect/group1
mkdir -p /var/zosconnect/group2
mkdir -p /var/zosconnect/group3
```

SYS1.PARMLIB (BPXPRM##)

```
MOUNT FILESYSTEM('OMVS.ZCEEVAR.ZFS')      ◀
  MOUNTPOINT('/var/zosconnect')
  TYPE(ZFS) MODE(READ)
MOUNT FILESYSTEM('OMVS.ZCEE.SERVERS.ZFS') ▶
  MOUNTPOINT('/var/zosconnect/servers')
  TYPE(ZFS) PARM('AGGRGROW') MODE(RDWR)
MOUNT FILESYSTEM('OMVS.ZCEE.GROUP1.ZFS') ◀
  MOUNTPOINT('/var/zosconnect/group1')
  TYPE(ZFS) PARM('AGGRGROW') MODE(RDWR)
MOUNT FILESYSTEM('OMVS.ZCEE.GROUP2.ZFS') ▶
  MOUNTPOINT('/var/zosconnect/group2')
  TYPE(ZFS) PARM('AGGRGROW') MODE(RDWR)
MOUNT FILESYSTEM('OMVS.ZCEE.GROUP.ZFS')
  MOUNTPOINT('/var/zosconnect/group3')
  TYPE(ZFS) PARM('AGGRGROW') MODE(RDWR)
```

- Create a dedicated filesystem for the root z/OS Connect /var directory, e.g., /var/zosconnect/v3r0/extensions. This directory structure can not be changed. This provides portability for migrations and system upgrades. Note: MODE(READ) will apply to /var/zosconnect/servers.

- Create a dedicated filesystem for each set or groups of servers. These filesystems will contain the server configuration directories for 1 or more servers.
- Each server's WLP_USER_DIR environment variable will be set to the mount point, e.g., *WLP_USER_DIR=/var/zosconnect/group1* when the server is created and in the server's startup JCL.

df -P | grep /var/zosconnect

Filesystem	512-blocks	Used	Available	Capacity	Mounted on
OMVS.ZCEEVAR.ZFS	69120	68658	462	100%	/var/zosconnect
OMVS.ZCEE.SERVERS.ZFS	159120	76455	82665	48%	/var/zosconnect/servers
OMVS.ZCEE.GROUP1.ZFS	135360	1506	133854	2%	/var/zosconnect/group1
OMVS.ZCEE.GROUP2.ZFS	4059360	2591284	1468076	64%	/var/zosconnect/group2
OMVS.ZCEE.GROUP3.ZFS	135360	17858	117502	14%	/var/zosconnect/group3



Let's review the z/OS Connect server configuration directory and files



ID=**LIBSERV**
Group=**LIBGRP**

```
export JAVA_HOME=<path_to_64_bit_Java>
export WLP_USER_DIR=/var/zosconnect
./zosconnect create zceesrvr
--template= zosconnect:apiRequester
```

/var/zosconnect	750	LIBSERV	LIBGRP
/servers	750	LIBSERV	LIBGRP
/zceesrvr	750	LIBSERV	LIBGRP
/apps	750	LIBSERV	LIBGRP
/configDropins	750	LIBSERV	LIBGRP
/overrides	750	LIBSERV	LIBGRP
/logs	777	LIBSERV	LIBGRP
messages.log	666	LIBSERV	LIBGRP
/resources	750	LIBSERV	LIBGRP
/zosconnect	750	LIBSERV	LIBGRP
/apis	750	LIBSERV	LIBGRP
/apiRequesters	750	LIBSERV	LIBGRP*
/rules	750	LIBSERV	LIBGRP
/services	750	LIBSERV	LIBGRP
/security	777	LIBSERV	LIBGRP
server.xml	640	LIBSERV	LIBGRP
/shared	750	LIBSERV	LIBGRP
/apps	750	LIBSERV	LIBGRP
/config	750	LIBSERV	LIBGRP

The **create** command will create the directories and files under the <**WLP_USER_DIR**> and assign ownership based on the ID and Group that created the server

There are a few potential issues with this in a production setting:

- If you have multiple people with a need to change configuration files, do you share the password of LIBSERV?

Sharing passwords is a bad practice. Better to take advantage SAF SURROGAT so permitted users can switch to the owning ID so they can make changes. In fact, LIBSERV should be a PROTECTED identity with no password in the first place.
- If you have multiple people with a need to read or update configuration files, do you simply connect them to LIBGRP?

The owner group may be granted access to other resources (on z/OS SAF profiles notably: SERVER) and you do not want others inheriting that. Better to make the configuration group be something different from the owner group and grant READ/WRITE through that group.
- The **shared** directory structure is shared among all servers started with a common value for the **WLP_USER_DIR** environment variable. Each server can access common server configuration files using the **shared.config.dir** environment variable or access web applications using the **shared.app.dir** environment variable.

* Only created when using the apiRequester template.



Using permission bits to control access



ID=**LIBSERV**
Group=**LIBGRP**

```
export JAVA_HOME=<path_to_64_bit_Java>
export WLP_USER_DIR=/var/zosconnect
./server create zceesrvr
```

```
/var/zosconnect          751 LIBSERV LIBGRP
  /servers               751 LIBSERV LIBGRP
    /zceesrv1            751 LIBSERV LIBGRP
      /apps                761 LIBSERV LIBGRP
      /configDropins       761 LIBSERV LIBGRP
        /overrides          761 LIBSERV LIBGRP
      /logs                 771 LIBSERV LIBGRP
        messages.log         644 LIBSERV LIBGRP
      /resources             751 LIBSERV ADMGRP
        /security            777 LIBSERV LIBGRP
        /zosconnect          751 LIBSERV ADMGRP
          /apis                761 LIBSERV ADMGRP
          /apiRequesters       761 LIBSERV ADMGRP
          /rules                761 LIBSERV ADMGRP
          /services              761 LIBSERV ADMGRP
        server.xml            460 LIBSERV ADMGRP
```

~~Often you may be tempted to use command **chmod -R 777 ***~~

Sample OMVS commands to manage permission bits

```
export WLP_USER_DIR=/var/zosconnect
cd $WLP_USER_DIR
chmod o+x -R servers
chmod o+x servers/zceesrvr/resources
chmod -R o+x servers/zceesrvr/resources/*
chmod g+r -R servers
chmod g+r servers/zceesrvr/resources
chmod -R g+r servers/zceesrvr/resources/*
chmod g+w server.xml
```

Warning: Access for Owner, Group(g), Others(o) depend on user ID (UID) and group ID (GID) as stored with the directory or file, not the actual SAF identity or group. This has implications when moving entire filesystems from one LPAR to another using utilities like ADRDSSU.

Tec-Tip: OMVS security - A quick review of Unix permissions bits

Owner	Group	Other																																																												
<table border="1"> <thead> <tr> <th>Bit</th> <th>Read</th> <th>Write</th> <th>Execute</th> </tr> </thead> <tbody> <tr> <td></td> <td>1</td> <td>1</td> <td>1</td> </tr> <tr> <td>Base-2 Value</td> <td>[4]</td> <td>[2]</td> <td>[1]</td> </tr> <tr> <td>↓</td> <td>↓</td> <td>↓</td> <td>↓</td> </tr> <tr> <td>4 + 2 + 1 =</td> <td></td> <td></td> <td></td> </tr> </tbody> </table> <p>7 The owner has READ, WRITE and EXECUTE</p>  <p>The owner of the file or directory</p> <pre>chmod -R * u+rwx zceesrv1</pre>	Bit	Read	Write	Execute		1	1	1	Base-2 Value	[4]	[2]	[1]	↓	↓	↓	↓	4 + 2 + 1 =				<table border="1"> <thead> <tr> <th>Bit</th> <th>Read</th> <th>Write</th> <th>Execute</th> </tr> </thead> <tbody> <tr> <td></td> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>Base-2 Value</td> <td>[4]</td> <td>[2]</td> <td>[1]</td> </tr> <tr> <td>↓</td> <td>↓</td> <td>↓</td> <td>↓</td> </tr> <tr> <td>4 + 0 + 1 =</td> <td></td> <td></td> <td></td> </tr> </tbody> </table> <p>5 The group has READ and EXECUTE, but not WRITE</p>  <p>IDs that are part of the group for the file or directory</p> <pre>chmod g+rwx server.xml</pre>	Bit	Read	Write	Execute		1	0	1	Base-2 Value	[4]	[2]	[1]	↓	↓	↓	↓	4 + 0 + 1 =				<table border="1"> <thead> <tr> <th>Bit</th> <th>Read</th> <th>Write</th> <th>Execute</th> </tr> </thead> <tbody> <tr> <td></td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>Base-2 Value</td> <td>[4]</td> <td>[2]</td> <td>[1]</td> </tr> <tr> <td>↓</td> <td>↓</td> <td>↓</td> <td>↓</td> </tr> <tr> <td>0 + 0 + 0 =</td> <td></td> <td></td> <td></td> </tr> </tbody> </table> <p>0 Others have nothing</p>  <p>IDs that are not the owner and not part of the group; that is, other</p> <pre>chmod -R * o+rx resources chmod -R * o-w resources/security</pre>	Bit	Read	Write	Execute		0	0	0	Base-2 Value	[4]	[2]	[1]	↓	↓	↓	↓	0 + 0 + 0 =			
Bit	Read	Write	Execute																																																											
	1	1	1																																																											
Base-2 Value	[4]	[2]	[1]																																																											
↓	↓	↓	↓																																																											
4 + 2 + 1 =																																																														
Bit	Read	Write	Execute																																																											
	1	0	1																																																											
Base-2 Value	[4]	[2]	[1]																																																											
↓	↓	↓	↓																																																											
4 + 0 + 1 =																																																														
Bit	Read	Write	Execute																																																											
	0	0	0																																																											
Base-2 Value	[4]	[2]	[1]																																																											
↓	↓	↓	↓																																																											
0 + 0 + 0 =																																																														

-R* indicates recursion



Tech-Tip: Use JCL to make the creation and configuration of servers repeatable and portable

And take advantage of RACF SURROGAT and UNIXPRIV resources

Example of using **SURROGAT** privileges

```
//ZCEESRVR JOB CLASS=A,REGION=0M,NOTIFY=&SYSUID,USER=LIBSERV
//*****
//**  SET SYMBOLS
//*****
//EXPORT EXPORT SYMLIST=(*)
// SET JAVAHOME='/usr/lpp/java/J8.0_64'
// SET ZCEEPATH='/usr/lpp/IBM/zosconnect/v3r0'
// SET SERVER='zceesRvr'
// SET TEMPLATE='zosconnect:default'
// SET WLPUSER='/var/ats/zosconnect'
// SET USER='ATSSERV'
// SET GROUP='ATSGRP'
//*****
//** Step ZCEESRVR - Use the zosconnect command to create a server
//*****
//ZCEESRVR EXEC PGM=IKJEFT01,REGION=0M
//SYSTSPRT DD SYSOUT=*
//SYSERR DD SYSOUT=*
//STDOUT DD SYSOUT=*
//SYSTSIN DD *,SYMBOLS=EXECSYS
BPXBATCH SH +
export JAVA_HOME=&JAVAHOME; +
export WLP_USER_DIR=&WLPUSER; +
&ZCEEPATH/bin/zosconnect create &SERVER +
--template=&TEMPLATE; +
```

Example of using **UNIXPRIV** privileges

```
//ZCEESRVR JOB CLASS=A,REGION=0M,NOTIFY=&SYSUID
//*****
//**  SET SYMBOLS
//*****
//EXPORT EXPORT SYMLIST=(*)
// SET JAVAHOME='/usr/lpp/java/J8.0_64'
// SET ZCEEPATH='/usr/lpp/IBM/zosconnect/v3r0'
// SET SERVER='openApi3'
// SET TEMPLATE='zosconnect:openApi3'
// SET WLPUSER='/var/ats/zosconnect'
// SET CONFIG='configDropins/overrides'
// SET USER='ATSSERV'
// SET GROUP='ATSGRP'
//*****
//** Step ZCEEAPI3 - Use the zosconnect command to create a server
//*****
//ZCEEAPI3 EXEC PGM=IKJEFT01,REGION=0M
//SYSTSPRT DD SYSOUT=*
//SYSERR DD SYSOUT=*
//STDOUT DD SYSOUT=*
//SYSTSIN DD *,SYMBOLS=EXECSYS
BPXBATCH SH +
export JAVA_HOME=&JAVAHOME; +
export WLP_USER_DIR=&WLPUSER; +
&ZCEEPATH/bin/zosconnect create &SERVER +
--template=&TEMPLATE; +
chown -R &USER:&GROUP $WLP_USER_DIR/servers/&SERVER
```

Tech/Tip: Use SAF SURROGAT resources for administration

RACF Surrogate access allows a designated administrative identity the ability to invoke commands and perform functions as if they were running under the identity that will be used for the z/OS Connect server started task. This may be useful because identities associated with started task are normally restricted and cannot be used for accessing TSO or OMVS shells,

Use the following examples as guides and create the surrogate resources and permit access. In these examples, ***LIBSERV*** represents the RACF identity under which the z/OS Connect server will be running and ***adminUser*** represent the administrative RACF identity.

Define a SURROGAT profile for the server's SAF identity

RDEFINE SURROGAT BPX.SRV.*LIBSERV*

Define a SURROGAT submit profile to allow job submission as the server's SAF identity

RDEFINE SURROGAT *LIBSERV*.SUBMIT

Permit an administrative identity to act as a surrogate of the Liberty task identity

PERMIT BPX.SRV.*LIBSERV* CLASS(SURROGAT) ID(*adminGrp*) ACC(READ)

PERMIT *LIBSERV*.SUBMIT CLASS(SURROGAT) ID(*adminGrp*) ACC(READ)

Refresh the SURROGAT in storage profiles

SETROPTS RACLIST(SURROGAT) REFRESH

Now any identity in group *adminGrp* can submit JCL with the *USER=LIBSERV* parameter on the job card or use the OMVS switch user command (*su -s LIBSERV*) to execute OMVS scripts or commands as LIBSERV.

Tech/Tip: z/OS : Also use SAF UNIXPRIV/FACILITY resources

An alternative to using a surrogate access is to permit the identity under which the customization will be done to enhanced Unix privileges. Specially, permitting the identity to Unix privileges SUPERUSER.FILESYS, SUPERUSER.FILESYS.CHANGEPERMS and SUPERUSER.FILESYS.CHOWN.

- *Permit an administrative identity to write to any local directory or file*
PERMIT SUPERUSER.FILESYS CLASS(UNIXPRIV)
 ID(adminUser) ACC(CONTROL)
- *Permit an administrative identity to change permission bit of any local directory or file*
PERMIT SUPERUSER.FILESYS.CHANGEPERMS CLASS(UNIXPRIV)
 ID(adminUser) ACC(READ)
- *Permit an administrative identity to change the ownership of any directory or file*
PERMIT SUPERUSER.FILESYS.CHOWN CLASS(UNIXPRIV)
 ID(adminUser) ACC(READ)
- *Permit an administrative identity switch to root (su -s root) or the Enable superuser mode(SU) Setup option in ISHELL*
PERMIT BPX.SUPERUSER CLASS(FACILITY) ID(adminUser) ACC(READ)
- *Refresh the UNIXPRIV and/or FACILITY instorage profiles*
SETROPTS RACLIST(UNIXPRIV,FACILITY) REFRESH

https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.4.0/com.ibm.zos.v2r4.bpxb200/usspriv.htm

Use the power of these commands provide carefully and only when necessary

A more detailed tour of a LPAR's directories and files



```
/var/zosconnect/v3r0<  
 /extensions (see previous slide)
```

```
 ${WLP_USER_DIR} <  
 /servers (details on next page)  
 /serverName
```

- The *extensions* subdirectory will always be in /var/zosconnect/v3r0

- There can be multiple \$WLP_USER_DIR directory on an LPAR
- Each server (serverName) will have a unique subdirectory in the location specified by WLP_USER_DIR, which **defaults** to /var/zosconnect.
- Important, use the same value for starting a server that was used when the server was created.

- The location of the *serverName* directory is based on the concatenation of the value of the *WLP_USER_DIR* environment variable with the constant *servers* and does not have to be in directory /var/zosconnect.
- The *serverName* directory structure and its initial contents are created by invoking the *zosconnect create serverName* script.
- serverName* can be a mount point with a dedicated file system mounted at this mount point (see above). This can be used to isolate servers to dedicated file systems.
- The number, size and output location of messages.log and trace files in the logs directory can be controlled with the Liberty <logging> configuration element or the output location controlled by using the *com.ibm.ws.logging.log.directory* Java directive as a JVM options override, more on this later.
- #These directories maintain state information and it is a good practice is to add the --clean parameter to the server startup JCL, e.g., PARMS='serverName --clean', especially after service is applied.



The contents of a server's configuration files

A server's configuration structure looks like this (N.B. OpenAPI 2 and OpenAPI 3 servers do not coexist as shown here):

```
$ {WLP_USER_DIR}
  /servers
    /serverName
      /apps
      /configDropins
        /overrides
      /logs
        /ffdc
        messages.log
      /resources
        /security
          /zosconnect
            /apis
            /apiRequesters
            /rules
            /services
          server.xml
        /tranlog
        /workarea
```

The `/apps` directory is the location to where OpenAPI 3 Web Archive (WAR) files are deployed.

The `/configDropins/overrides` directory is the location where server XML configuration files are placed for OpenAPI 3 servers.

The `messages.log` file is the key output file for messages about Liberty and the processing taking place in the Liberty server. The output written to this file can be written to the SPOOL by including DD statement MSGLOG in the startup JCL, e.g., //MSGLOG DD SYSOUT=*,FREE=CLOSE,SPIN=(UNALLOC,1M)

The `/security` directory contains files `ltpa.keys` and `key.p12`. `ltpa.keys` is the server specific LTPA token. `key.p12` is a self-signed certificate that expires in one year.

The `/zosconnect` directory is where the deployed APIs, services, and API requester files will be placed for an OpenAPI 2 server.

The `server.xml` file is the key configuration file. It is here that z/OS Connect definitions go which define the essential configuration of the server and backend connectivity.

The `WLP_USER_DIR` environment variable sets the value of the root directory of the server's configuration files and directories, e.g.,
`WLP_USER_DIR=/var/zosconnect`



Tech-Tip: Use “include” files to extend and manage a server’s configuration

- Setup a server.xml using ‘include’ statements and allow other administrator to manage those included files, but not the server.xml itself.
- Control what configuration can be overridden in included files using the ‘onConflict’ option provided with the include element (see Ignore, Replace, Merge).

https://www.ibm.com/support/knowledgecenter/en/SSAW57_liberty/com.ibm.websphere.wlp.nd.multiplatform.doc/ae/cwlp_config_include.html

server.xml (owned by ID ADMIN1)

```
<include location="${server.config.dir}/includes/db2.xml onConflict="IGNORE"/>
<include location="${server.config.dir}/includes/cics.xml onConflict="IGNORE"/>
<include location="${server.config.dir}/includes/imsDb.xml onConflict="IGNORE"/>
<featureManager>
  <feature>zosconnect:zosConnect-2.0</feature>
  <feature>zosconnect:zosConnectCommands-1.0</feature>
  <feature>apiDiscovery-1.0</feature>
<featureManager>
```

db2.xml (owned and managed by a DBA)

```
<server description="Db2 REST">
  <zosconnect_zosConnectServiceRestClientConnection
    id="Db2Conn" host="wg31.washington.ibm.com" port="2446" basicAuthRef="dsn2Auth" />
  <zosconnect_zosConnectServiceRestClientBasicAuth id="dsn2Auth"
    applName=DSN2APPL/>
</server>
```

cics.xml (owned and managed by a CICS administrator)

```
<server description="CICS">
  <featureManager> <feature>zosconnect:cicsService-1.0</feature> </featureManager>
  <zosconnect_cicsIpicConnection id="catalog" host="wg31" port="1491"/>
  <zosconnect_cicsIpicConnection id="cscvinc" host="wg31" port="1493"/>
</server>
```

imsDB.xml (owned and managed by a IMS administrator)

```
<server description="IMS DATABASE">
  <featureManager> <feature>zosconnect:dbService-1.0</feature> </featureManager>
  <connectionFactory id="DFSIVPACConn" > <properties:imsudbJLocat
    databaseName="DFSIVPA" datastoreName="IVP1" driverType="4" portNumber="5555"
    datastoreServer="wg31" user="USER1" password="USER1"
    flattenTables="True"/> </connectionFactory>
</server>
```

Nesting of an include file within a include file is possible



Tech-Tip: Review configuration conflicts

```
ÝAUDIT  " CWWKG0102I: Found conflicting settings for cscvincAPI instance of zosconnect_endpointConnection configuration.  
Property port has conflicting values:  
  Value 9443 is set in file:/var/zosconnect/servers/myServer/includes/apiRequesterHTTPS.xml.  
  Value 9443 is set in file:/var/zosconnect/servers/myServer/includes/apiRequesterHTTPS.xml.  
  Value 9463 is set in file:/var/zosconnect/servers/myServer/includes/oauth.xml.  
Property port will be set to 9463.  
Property host has conflicting values:  
  Value https://dvipa.washington.ibm.com is set in file:/var/zosconnect/servers/myServer/includes/apiRequesterHTTPS.xml.  
  Value https://dvipa.washington.ibm.com is set in file:/var/zosconnect/servers/myServer/includes/apiRequesterHTTPS.xml.  
  Value https://mpz3.washington.ibm.com is set in file:/var/zosconnect/servers/myServer/includes/oauth.xml.  
Property host will be set to https://mpz3.washington.ibm.com.  
Property authenticationConfigRef has conflicting values:  
  Value mySAFAuth is set in file:/var/zosconnect/servers/myServer/includes/apiRequesterHTTPS.xml.  
  Value myoAuthConfig is set in file:/var/zosconnect/servers/myServer/includes/oauth.xml.  
Property authenticationConfigRef will be set to myoAuthConfig.
```

onConflict="MERGE" Conflicting elements will be merged, and the last value encountered will be used.

onConflict="REPLACE" When elements conflict, the element in the included file will be ignored

onConflict="IGNORE" Conflicting elements in the included file are ignored.



Use the *bootstrap.properties* file to help customize a server's XML configuration[#]

zceesrv1's bootstrap.properties

```
httpPort=9080  
httpsPort=9443  
ipicPort=1491  
host=*  
cicsHost=wg31.washington.ibm.com  
network=ZOSCONN1  
applid=ZOSCONN1  
com.ibm.ws.zos.core.angelName=namedAngel
```

Java directives can also be provided.

zceesrv2's bootstrap.properties

```
httpPort=9090  
httpsPort=9453  
ipicPort=1492  
host=wg31.washington.ibm.com  
cicsHost=wg31.washington.ibm.com  
network=ZOSCONN2  
applid=ZOSCONN2  
com.ibm.ws.zos.core.angelName=namedAngel
```

server.xml

```
<!-- To access this server from a remote client, add a host attribute to the following  
element, e.g. host="*" -->  
    <httpEndpoint id="defaultHttpEndpoint"  
        host="${host}"  
        httpPort="${httpPort}"  
        httpsPort="${httpsPort}" />
```

ipicIDProp.xml

```
<zosconnect_cicsIpicConnection id="catalog"  
    host="${cicsHost}" port="${ipicPort}"  
    zosConnectNetworkid="${network}" zosConnectApplid="${applid}"/>  
  
<zosconnect_cicsIpicConnection id="cscvinc"  
    host="${cicsHost}" port="${ipicPort}"  
    zosConnectNetworkid="${network}" zosConnectApplid="${applid}"/>  
  
<zosconnect_cicsIpicConnection id="miniloan"  
    host="${cicsHost}" port="${ipicPort}"  
    zosConnectNetworkid="${network}" zosConnectApplid="${applid}"/>
```

#Located in directory
\${server.config.dir} and
uses EBCDIC encoding

Tech-Tip: A suggestion for managing the server.xml configuration file



Default server.xml configuration file

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="new server">
    <!-- Enable features -->
    <featureManager>
        </featureManager>
        <!-- To access this server from a remote client add a host attribute -->
        <httpEndpoint id="defaultHttpEndpoint">
            host="*"
            httpPort="9080"
            httpsPort="9443" />
    <!-- add cors to allow cross origin access, e.g. when using swagger -->
    <cors id="defaultCORSConfig">
        domain="/"
        allowedOrigins="*"
        allowedMethods="GET, POST, PUT, DELETE, OPTIONS"
        allowedHeaders="Origin, Content-Type, Authorization, Cache-Control"
        allowCredentials="true"
        maxAge="3600"/>
    </server>
```

Modified server.xml configuration file

The simplifies administration by :

- Using a *bootstrap.properties* file to customize the ports in the *server.xml* file.
 - Using “include” statements to make further changes such as adding additional features and additional XML configuration elements.
 - Review <https://www.ibm.com/docs/en/was-liberty/nd?topic=liberty-configuration-element-merging-rules> to understand merging rules.
 - **Consider providing configuration elements by placing server XML files in the .../configDropins/original subdirectory.**

A Liberty JCL procedure versus a z/OS Connect started task JCL procedure

```
//ZCEESRVR PROC PARMs='zceesrvr'  
/*  
// SET ZCONHOME='/usr/lpp/IBM/zosconnect/v3r0'  
// SET INSTDIR='/usr/lpp/liberty_zos/21.0.0.9'  
/*  
//ZCON      EXEC PGM=BPXBATSL,REGION=0M,MEMLIMIT=8G,  
//   PARM='PGM &ZCONHOME./bin/zosconnect run &PARMS. --clean'  
//   PARM='PGM &INSTDIR./lib/native/zos/s390x/bbgzsrv &PARMS'  
//STDOUT    DD     SYSOUT=*  
//STDERR    DD     SYSOUT=*  
//STDIN     DD     DUMMY  
//STDENV    DD     *  
_BPX_SHAREAS=YES  
_CEE_RUNOPTS=HEAPPOOLS (ON) ,HEAPPOOLS64 (ON)  
JAVA_HOME=/usr/lpp/java/J8.0_64  
WLP_USER_DIR=/var/zosconnect  
JVM_OPTIONS=-Dcom.ibm.ws.zos.core.angelName=ZCEE -Xmx512m  
OPENJ9_JAVA_OPTIONS=-Xoptionsfile=/var/zcee/properties/myServer.property
```

OMVS
LE
JAVA
LIBERTY
z/OS Connect



An example of providing STDENV input in a JCL Procedure

Use the STDENV DD statement to scale servers and share configuration properties horizontally

```
//ZCON      EXEC PGM=BPXBATSL,REGION=0M,MEMLIMIT=8G,  
//                                PARM='PGM &ZCONHOME./bin/zosconnect run &PARMS. --clean'  
//STDERR    DD SYSOUT=*,FREE=CLOSE,SPIN=(UNALLOC,1M)  
//STDOUT    DD SYSOUT=*  
//STDIN     DD DUMMY  
//STDENV    DD PATH='/var/zcee/properties/&PARMS..property',  
//                           PATHOPTS=ORDONLY  
//  
// or  
//STDENV    DD DISP=SHR,DSN=JOHNSON.ZCEE.STDENV(COMMON)  
//  
//          DD DISP=SHR,DSN=JOHNSON.ZCEE.STDENV(ZCEESRVR)
```

Either one OMVS property file or multiple PDS members.

The last occurrence environment variable encountered determines the value of the environment variable.

```
Member COMMON  
_BPX_SHAREAS=YES  
_CEE_RUNOPTS=HEAPPOOLS(ON),HEAPPOOLS64(ON)  
JAVA_HOME=/usr/lpp/java/J8.0_64  
ZCON_ENV_DEBUG=TRUE  
WLP_USER_DIR=/var/zosconnect
```

Which value used for a Java option or property depends on which environment variable is used to specify the option or property.

```
Member ZCEESRVR  
OPENJ9_JAVA_OPTIONS=-Dcom.ibm.ws.zos.core.angelName=ZCEEANGL  
JVM_OPTIONS=-Xoptionsfile=/var/zcee/properties/javaHCD.property -Xmx512m -verbose:sizes  
JAVA_HOME=/u/johnson/java/J8.0_64  
WLP_USER_DIR=/var/ats/zosconnect
```

Using //SYSIN DD * is discouraged because of the 80 character record limit.



STDENV DD concatenation and environment variables precedence order

Member COMMON

```
_BPX_SHAREAS=YES  
_CEE_RUNOPTS=HEAPPOOLS (ON) ,HEAPPOOLS64 (ON)  
JAVA_HOME=/usr/lpp/java/J8.0_64  
ZCON_ENV_DEBUG=TRUE  
WLP_USER_DIR=/var/alt/zosconnect
```

Green indicated the environment variable, Java option(-X) or system property(-D) that are used.
Red indicates the environment variable, Java option(-X) or system property(-D) that are ignored.

Member OPENJ9

```
OPENJ9_JAVA_OPTIONS=-verbose:sizes -Xms75m -Dcom.ibm.ws.zos.core.angelName=OPENJ9  
-Dcom.ibm.ws.logging.message.file.name=openj9.log #
```

Member IBMOPTS

```
IBM_JAVA_OPTIONS=-verbose:jni -Xms80m -Dcom.ibm.ws.logging.message.file.name=ibmopts.log  
-Dcom.ibm.ws.zos.core.angelName=IBMOPTS #
```

Member JVMOPTHC

```
JVM_OPTIONS=-Xoptionsfile=/var/zcee/properties/javaHCD.property -Dcom.ibm.ws.zos.core.angelName= -Xmx256m -verbose:sizes
```

Member JAVAHOME

```
JAVA_HOME=/u/johnson/java/J8.0_64
```

Member ZCEEANGL

```
OPENJ9_JAVA_OPTIONS=-Dcom.ibm.ws.zos.core.angelName=ZCEEANGL -Dcom.ibm.ws.logging.message.file.name=zceeanogl.log -Xmx16m -Xms60m  
-verbose:gc #
```

Member WLPUSER

```
WLP_USER_DIR=/var/zosconnect
```

Default settings for the OpenJ9 VM <https://www.ibm.com/docs/en/sdk-java-technology/8?topic=reference-default-settings>



Key Liberty, z/OS Connect and Java environment variables

- **WLP_USER_DIR** – This environment variable is used when a server is created to determine where the server's working directories will be created and where the initial *server.xml* file will be created. This variable is also used by the runtime environment to locate the server's existing working directories and the *server.xml* file. Also, the **WLP_USER_DIR** is used to set the shared variables..

The following environment variables are automatically set in a Liberty server and can be used as variables in the server XML configuration files.

- **server.config.dir** – whose value will automatically be set to the value of variable *WLP_USER_DIR* concatenated with the name of the server, e.g. */var/zosconnect/servers/serverName*
- **shared.config.dir** – whose value will automatically be set to the value of variable *WLP_USER_DIR* concatenated with /shared/config, e.g. */var/zosconnect/shared/config*
- **shared.app.dir** – whose value will automatically be set to the value of variable *WLP_USER_DIR* concatenated with /shared/apps, e.g. */var/zosconnect/shared/apps*
- **wlp.server.name** - whose value will automatically be set to the value of the server as provided in the *zosconnect run* command, e.g., PARMS value provided in the JCL procedure.

Java related environment variables are also available for use in a z/OS Connect environment.

- **JAVA_HOME** – The OMVS directory where the Java executables (*/bin* directory) can be located.
- **JVM_OPTIONS** – A z/OS Connect environment variables that provides Java options and/or system properties. The contents of **JVM_OPTIONS** is added to the *java* command line in the *zosconnect* startup script.
- **IBM_JAVA_OPTIONS** – An IBM JAVA environment variable (deprecated and eventually will be replaced by *environment variable OPENJ9_JAVA_OPTIONS*). Environment variable **IBM_JAVA_OPTIONS** variable can be used to provide Java options and/or system properties.
- **OPENJ9_JAVA_OPTIONS** – An OpenJ9 environment variable (eventually will replace the deprecated environment variable **IBM_JAVA_OPTIONS**). Environment variable **OPENJ9_JAVA_OPTIONS** variable can be used to provide Java options and/or system properties.

Note: Any Java option or system property using **JVM_OPTIONS** supersedes the same Java non-standard options or system property when provided by **IBM_JAVA_OPTIONS** or **OPENJ9_JAVA_OPTIONS**



Tech/Tip: Liberty Java Directives for controlling output

`com.ibm.ws.logging.console.format (consoleFormat)` - The required format for the console. Valid values are basic or json format.

`com.ibm.ws.logging.console.log.level (consoleLogLevel)` - This filter controls the granularity of messages that go to the console. The valid values are INFO, AUDIT, WARNING, ERROR, and OFF. By default, the console log level is set to AUDIT.

`com.ibm.ws.logging.hideMessage (hideMessage)` - Use this attribute to configure the messages that you want to hide from the `console.log` and `message.log` files. If the messages are configured to be hidden, then they are redirected to the `trace.log` file.

`com.ibm.ws.logging.log.directory (logDirectory)` - Use this attribute to set a directory for all log files, excluding the `console.log` file, but including FFDC. The default log location path is `WLP_OUTPUT_DIR/serverName/logs`

`com.ibm.ws.logging.max.file.size (maxFileSize)` - The maximum size (in MB) that a log file can reach before it is rolled. The Liberty runtime does only size-based log rolling. To disable this attribute, set the value to 0. The maximum file size is approximate. By default, the value is 20.

`com.ibm.ws.logging.max.files (maxFiles)` - If a maximum file size exists, this setting is used to determine how many of each of the log files are kept. This setting also applies to the number of exception logs that summarize exceptions that occurred on any day. So, if this number is 10, you might have 10 message logs, 10 trace logs, and 10 exception summaries in the `ffdc` directory. The default value is 2.

`com.ibm.ws.logging.message.format (messageFormat)` - The required format for the `messages.log` file. Valid values are basic or json format. By default, `messageFormat` is set to the environment variable `WLP_LOGGING_MESSAGE_FORMAT` (if set) or basic.

`com.ibm.ws.logging.trace.file.name (traceFileName)` - The `trace.log` file is only created if additional or detailed trace is enabled. `stdout` is recognized as a special value; and causes trace to be directed to the original standard out stream.

bootstrap.properties example:

```
com.ibm.ws.logging.message.file.name=basqstrtMessages.log  
com.ibm.ws.logging.log.directory=/u/common/logs
```

N.B. `consoleFormat`, `logDirectory`, etc. can be specified in the `<logging/>` Liberty configuration element. Note the recommendation for the attributes in red is for them to be provided in Java directives.

Tech-Tip: Consider using symbolic links especially as an administrative shortcut

- Create an “administration” subdirectory, e.g., `zcee` in directory `/var`
- Then create a symbolic link in the “administration” directory to each Liberty server’s configuration directory and other frequently accessed directories.

```
ls -al /var/zcee
drwxrwxrwx 4 JOHNSON SYS1          8192 Aug 16 12:23 .
drwxrwxrwt 25 OMVSKERN SYS1         8192 Aug 16 11:56 ..
lrwxrwxrwx 1 JOHNSON SYS1          57 Aug 16 12:22 CSCWLW -> /var/wlp/cics/CICS53Z/CSCWLW/wlp/usr/servers/defaultServer
lrwxrwxrwx 1 JOHNSON SYS1          57 Aug 16 12:22 CICSWLW -> /var/wlp/cics/CICS53Z/CICSWLW/wlp/usr/servers/cicswl
drwxrwxrwx 2 JOHNSON SYS1         8192 Aug 16 15:30 hcd
lrwxrwxrwx 1 JOHNSON SYS1          27 Jun 10 15:55 includes -> /global/zosconnect/includes
lrwxrwxrwx 1 JOHNSON SYS1          28 Aug 16 10:12 mqweb -> /var/mqm/mqweb/servers/mqweb
lrwxrwxrwx 1 JOHNSON SYS1          32 Jun  4 12:49 myServer -> /var/zosconnect/servers/myServer
drwxr-xr-x 2 JOHNSON SYS1         8192 Aug 16 13:14 properties
lrwxrwxrwx 1 JOHNSON SYS1          18 Aug 17 12:47 shared -> /var/shared/zosconnect/resources/zosconnect
lrwxrwxrwx 1 JOHNSON SYS1          24 May 13 2020 walop3a -> /var/wlp/servers/walop3a
lrwxrwxrwx 1 JOHNSON SYS1          24 May 13 2020 walrp3a -> /var/wlp/servers/walrp3a
lrwxrwxrwx 1 JOHNSON SYS1          31 May 14 2020 wazz34a -> /var/zosconnect/servers/wazz34a
lrwxrwxrwx 1 JOHNSON SYS1          24 Aug 16 10:32 wlphats -> /var/wlp/servers/wlphats
lrwxrwxrwx 1 JOHNSON SYS1          36 Aug 16 10:31 zceepir -> /var/ats/zosconnect/servers/zceepir
lrwxrwxrwx 1 JOHNSON SYS1          39 Aug 16 10:18 zceecics -> /var/cicsts/zosconnect/servers/zceecics
lrwxrwxrwx 1 JOHNSON SYS1          35 Aug 16 10:31 zceedvm -> /var/ats/zosconnect/servers/zceedvm
lrwxrwxrwx 1 JOHNSON SYS1          32 Jun 10 15:54 zceeoipid -> /var/zosconnect/servers/zceeoipid
lrwxrwxrwx 1 JOHNSON SYS1          36 Aug 16 10:14 zceesrvr -> /var/ats/zosconnect/servers/zceesrvr
lrwxrwxrwx 1 JOHNSON SYS1          44 Aug 16 11:57 zosmfServer -> /var/zosmf/configuration/servers/zosmfServer
```

Not all these directories are for z/OS Connect servers, there are CICS Liberty servers, a MQ Web Console Liberty server, a zOSMF Liberty server, a HATS Liberty server and a couple of standard Liberty servers for Java applications.

One administration directory to manage them all!

Administrative – Again use dedicated ZFS filesystem at the mount points

- Create mount points in the “administrative” directory for shared r/w directories
- Avoid creating directories and files in the root file system.
- Use a common or shared mount point
 - Use /var mount point for local read/write file systems
 - Use /global for sharing a mount point across multiple LPARs
- Use ZFS filesystems and use AGGRGROW to allow R/W ZFS filesystems to automatically go into extents (>16).

```
SYS1.PARMLIB(BPXPRM##)
MOUNT FILESYSTEM('OMVS.ZCEE.ZFS')
  MOUNTPOINT('/var/zcee')
  TYPE(ZFS) PARM('AGGRGROW') MODE(RDWR)
MOUNT FILESYSTEM('OMVS.ZCEEHCD.ZFS')
  MOUNTPOINT('/var/zcee/hcd')
  TYPE(ZFS) PARM('AGGRGROW') MODE(RDWR)
MOUNT FILESYSTEM('OMVS.ZCEE.SHARED.ZFS')
  MOUNTPOINT('/var/shared/zosconnect')
  TYPE(ZFS) PARM('AGGRGROW') MODE(RDWR)
```



Sharing XML configuration files between servers using \${shared.config.dir}

Add an “includes” subdirectories \${shared.config.dir} with a symbolic links to a common location. This common directory can be accessed from multiple servers on a single or from multiple LPARs. Additions and updates to the “include” files are then made in one single administrative directory.

Symbolic links from servers shared configuration \${shared.config.dir} to common directory

Symbolic links to a shared local LPAR directory

```
ln -s /var/shared/zosconnect/includes /var/zosconnect/shared/config  
ln -s /var/shared/zosconnect/includes /var/ats/zosconnect/shared/config  
ln -s /var/shared/zosconnect/includes /var/wsc/zosconnect/shared/config
```

Symbolic links to a shared Sysplex directory *

```
ln -s /global/zosconnect/includes /var/zosconnect/servers/shared/config  
ln -s /global/zosconnect/includes /var/ats/zosconnect/shared/config  
ln -s /global/zosconnect/includes /var/wsc/zosconnect/shared/config
```

The server.xml file contains these “include” statements

```
<include location="${shared.config.dir}/safSecurity.xml"/>  
<include location="${shared.config.dir}/ipicIDProp.xml"/>  
<include location="${shared.config.dir}/keyringOutboundMutual.xml"/>  
<include location="${shared.config.dir}/groupAccess.xml"/>  
<include location="${shared.config.dir}/shared.xml"/>  
<include location="${shared.config.dir}/db2.xml"/>  
<include location="${shared.config.dir}/oauth.xml"/>
```



/var/shared/zosconnect/includes

Contents of the common “includes” directory

*basicSecurity.xml
db2.xml
db2TLS.xml
groupAccess.xml
ipic.xml
ipicIDProp.xml
keyringInbound.xml
keystore.xml
keyringMutual.xml
keyringOutboundMutual.xml
safSecurity.xml*

For example, changing *basicSecurity.xml* to *safSecurity.xml* and refreshing the configuration changes security from basic to SAF

F ZCEESRVR ,REFRESH,CONFIG



A practical example-PTF V3.0.35 included a CORS update

July 2020

V3.0.35 (APAR PH26291)
Server code update

Enhancements

- The text of messages BAQR0417W and BAQR0418W has been updated. For more information, see z/OS Connect EE [Runtime Messages](#).

Fixes

- PH21761 A CICS region reports **SOS DFHSM0133 WBSEBUF** when z/OS Connect EE requester is in use.
- PH25345 Passing user credentials in the request body to the authentication server to obtain a JWT causes a NPE in z/OS Connect EE.
- PH21819 z/OS Connect EE sets some CORS headers automatically.

Attention

When this fix is applied, additional CORS configuration is required in `server.xml` to enable connections from the z/OS Connect EE API toolkit and JavaScript clients. For more information, see [Configuring Cross-Origin Resource Sharing on a z/OS Connect Enterprise Edition Server](#)

`cors.xml`

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="CORS entries">

    <!-- add cors to allow cross origin access, e.g. when using swagger doc from zOS Connect Enterprise
        Edition -->
    <cors id="defaultCORSConfig"
        domain="/"
        allowedOrigins="*"
        allowedMethods="GET, POST, PUT, DELETE, OPTIONS"
        allowedHeaders="Origin, Content-Type, Authorization, Cache-Control, Expires, Pragma"
        allowCredentials="true"
        maxAge="3600"/>

</server>
```

`server.xml`

```
<include location="${server.config.dir}/cors.xml"/>
```



Sharing XML configuration files – using '*variables*' files

“variables” files whose names are based on the name of the server

myServer.xml

```
<variable name= "unauthenticatedUser" value= "WSGUEST" />
<variable name="profilePrefix" value= "BBGZDFLT" />
```

zceoepid.xml

```
<variable name= "unauthenticatedUser" value="ZCGUEST" />
<variable name="profilePrefix" value="EMJZDFLT" />
```

server.xml

```
<server description="new server">
<include location="${server.config.dir}/includes/safSecurity.xml"/>
<include location="${server.config.dir}/includes/${wlp.server.name}.xml"/>

    <!-- Enable features -->
    <featureManager>
        <feature>zosconnect:zosConnect-2.0</feature>
        <feature>zosconnect:zosConnectCommands-1.0</feature>
    </featureManager>
```

safSecurity.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="SAF security">

    <!-- Enable features -->
    <featureManager>
        <feature>appSecurity-2.0</feature>
        <feature>zosSecurity-1.0</feature>
    </featureManager>

    <webAppSecurity allowFailOverToBasicAuth="true" />
    <safRegistry id="saf" />
    <safAuthorization racRouteLog="ASIS" />
    <safCredentials unauthenticatedUser="${unauthenticatedUser}"
        profilePrefix="${profilePrefix}" />
</server>
```



Using the Liberty server's configuration drop-in's directories

Located in the same directory as the *server.xml* configuration file.

- Configuration files in the */overrides* directory adds to or replaces the configuration elements found in *server.xml*
- Configuration files in the */default* directory provides defaults for configuration elements not present in *server.xml*

```
 ${WLP_USER_DIR}
  /servers
    /serverName
      /apps
      /configDropins
        /overrides
        /default
    /logs
      /ffdc
      messages.log
    /resources
      /security
      /zosconnect
      /apis
      /apiRequesters
      /rules
      /services
    server.xml
  /tranlog
  /workarea
```

```
commonFeatures.xml
<server description="Common Server Features">

  <!-- Enable features -->
  <featureManager>
    <feature>adminCenter-1.0</feature>
    <feature>restConnector-2.0</feature>
  </featureManager>

  <remoteFileAccess>
    <readDir>/var/zcee/includes</readDir>
    <readDir>/global/zosconnect/includes</readDir>
    <writeDir>${server.config.dir}</writeDir>
  </remoteFileAccess>

</server>
```

```
safSecurity.xml
<?xml version="1.0" encoding="UTF-8"?>
<server description="SAF security">

  <!-- Enable features -->
  <featureManager>
    <feature>appSecurity-2.0</feature>
    <feature>zosSecurity-1.0</feature>
  </featureManager>

  <webAppSecurity allowFailOverToBasicAuth="true" />
  <safRegistry id="saf" />
  <safAuthorization racRouteLog="ASIS" />
  <safCredentials unauthenticatedUser="${unauthenticatedUser}"
    profilePrefix="${profilePrefix}" />
</server>
```

Another directory that must be manually created.



Simplifying administration by combining include files and by using server variables

Default server.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="new server">
<include location="${server.config.dir}/includes/${wlp.server.name}.xml"/>

    <!-- Enable features -->
    <featureManager>
        <feature>zosconnect:zosConnect-3.0</feature>
        <feature>openapi-3.0</feature>
    </featureManager>

    <!-- To access this server from a remote client add a host attribute
        to the following element, e.g. host="*"
    -->
    <httpEndpoint id="defaultHttpEndpoint">
        host="*"
        httpPort="9080"
        httpsPort="9443" />
```

`${server.config.dir}/includes/${wlp.server.name}.xml`

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="new server">
<variable name="httpPort" value="9081"/>
<variable name="httpsPort" value="9445"/>
<variable name="hostName" value="*"/>
<variable name="CICS_HOST" value="wg31.washington.ibm.com"/>
<variable name="CICS_PORT" value="1491"/>
<variable name="DB2_HOST" value="wg31.washington.ibm.com"/>
<variable name="DB2_PORT" value="2446"/>
<variable name="DB2_USERNAME" value="USER2"/>
<variable name="DB2_PASSWORD" value="USER2"/>
<include location="${shared.config.dir}/safSecurity.xml"/>
<include location="${shared.config.dir}/httpEndpoint.xml"/>
<include location="${shared.config.dir}/db2.xml"/>
<include location="${shared.config.dir}/cics.xml"/>
<include location="${shared.config.dir}/keystore.xml"/>
</server>
```

```
 ${server.config.dir}/includes/httpEndpoint.xml"/>
<server description="basic security">
    <httpEndpoint id="defaultHttpEndpoint"
        host="${hostName}"
        httpPort="${httpPort}"
        httpsPort="${httpsPort}" />
</server>
```

`${server.config.dir}/includes/db2.xml`

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="Default server">
<featureManager>
    <feature>zosconnect:db2-1.0</feature>
</featureManager>
<zosconnect_credential user="${DB2_USERNAME}"
    password="${DB2_PASSWORD}" id="commonCredentials" />
<zosconnect_db2Connection id="db2Conn" host="${DB2_HOST}"
    port="${DB2_PORT}" credentialRef="commonCredentials" />
</server>
```

`${server.config.dir}/includes/cics.xml`

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="CICS IPIC connections">
    <!-- Enable features -->
    <featureManager>
        <feature>zosconnect:cics-1.0</feature>
    </featureManager>
    <zosconnect_cicsIpicConnection id="cicsConn" host="${CICS_HOST}"
        port="${CICS_PORT}" />
</server>
```

Use Symbolic links to simplify commands used in OMVS and JCL

Performing commands:

```
ln -s /global/zosconnect/includes /var/zcee/includes
ln -s /var/zosconnect/servers/zceesrv1 /var/zcee/zceesrv1
ln -s /var/zosconnect/servers/zceesrv2 /var/zcee/zceesrv2
```

Will change these OMVS commands from:

```
ln -s /global/zosconnect/includes /var/zosconnect/servers/zceesrv1/includes
ln -s /global/zosconnect/includes /var/zosconnect/servers/zceesrv2/includes
```

To simpler (and shorter) OMVS commands:

```
ln -s /var/zcee/includes /var/zcee/zceesrv1/includes
ln -s /var/zcee/includes /var/zcee/zceesrv2/includes
```

Directory Shortcuts

- Create a shortcut from the shared administrative *include* directory to the Sysplex or LPAR shared directory
- Create shortcuts from the server's administrative directories to each server's configuration directory.

N.B. These are symbolic links to symbolic links.

ln -s oldname newname

These symbolic links can be used as JCL symbols

```
//EXPORT EXPORT SYMLIST=(*)
// SET SERVER= 'zceesrv1'
// SET SHARED=' /var/zcee/includes '
// SET WLPUSER=' /var/zosconnect '
//ZCEELN EXEC PGM=IKJEFT01,REGION=0M
//SYSTSPRT DD SYSOUT=*
//SYSERR DD SYSOUT=*
//STDOUT DD SYSOUT=*
//SYSTSIN DD *,SYMBOLS=EXECSYS
BPXBATCH SH +
ln -s &SHARED /var/zcee/&SERVER/includes
instead of entering the full directory names as in
ln -s /global/zosconnect/includes +
&WLPUSER/servers/&SERVER/includes
```

And added as exports to /u/home/.profile or /etc/profile files

```
export serverName=zceesrv1
export shared=/var/zcee/includes
export WLP_USER_DIR=/var/zosconnect
```

```
//ZCEELN EXEC PGM=IKJEFT01,REGION=0M
//SYSTSPRT DD SYSOUT=*
//SYSERR DD SYSOUT=*
//STDOUT DD SYSOUT=*
//SYSTSIN DD *
BPXBATCH SH +
ln -s $shared /var/zcee/$serverName/includes
instead of entering the full directory names as in
ln -s /global/zosconnect/includes +
$WLPUSER/servers/$serverName/includes
```



Use JCL to make the creation and configuration of servers repeatable and portable

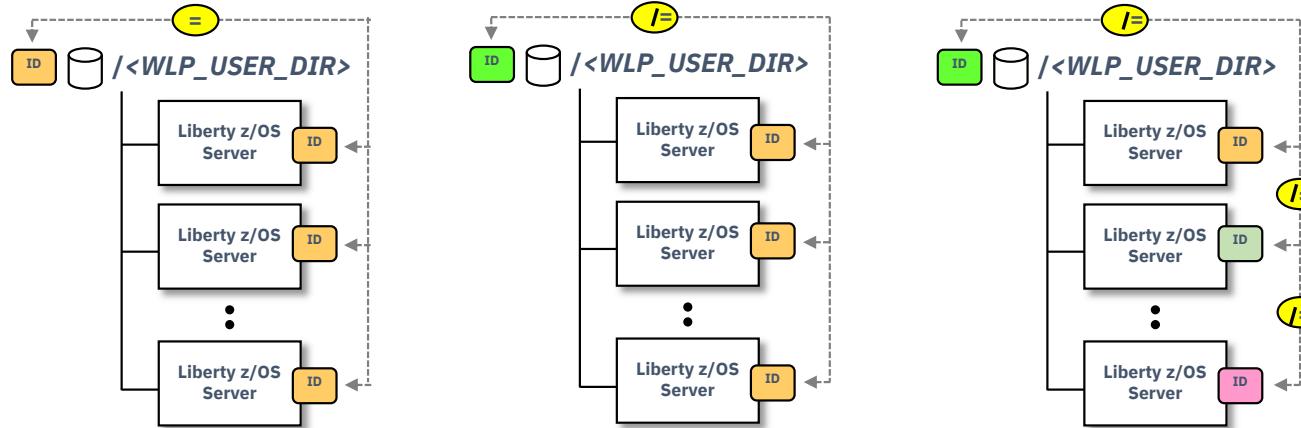
```
//*****
//* SET SYMBOLS
//*****
//EXPORT EXPORT SYMLIST=(*)
// SET JAVAHOME='/usr/lpp/java/J8.0_64'
// SET ZCEEPATH='/usr/lpp/IBM/zosconnect/v3r0'
// SET SERVER='zceesrvr'
// SET TEMPLATE='zosconnect:default'
// SET WLPUSER='/var/ats/zosconnect'
// SET USER='ATSSERV'
// SET GROUP='ATSGRP'
//*****
//** Step ZCEESRVR - Use the zosconnect command to create a server
//*****
//ZCEESRVR EXEC PGM=IKJEFT01,REGION=0M
//SYSTSPRT DD SYSOUT=*
//SYSERR DD SYSOUT=*
//STDOUT DD SYSOUT=*
//SYSTSIN DD *,SYMBOLS=EXEC SYS
BPXBATCH SH +
export JAVA_HOME=&JAVAHOME; +
export WLP_USER_DIR=&WLPUSER; +
&ZCEEPATH/bin/zosconnect create &SERVER +
--template=&TEMPLATE; +
ln -s $WLP_USER_DIR/servers/&SERVER /var/zceesrvr/&SERVER; +
ln -s /var/shared/includes/commonFeatures.xml +
 /var/zceesrvr/&CONFIG/commonFeatures.xml; +
ln -s /var/shared/includes/cors.xml +
 /var/zceesrvr/&CONFIG/cors.xml; +
ln -s /var/shared/includes/safSecurity.xml +
 /var/zceesrvr/&CONFIG/safSecurity.xml; +
cp /var/zceesrvr/properties/bootstrap.properties +
 /var/zceesrvr; +
cp /var/zceesrvr/properties/server.xml +
 /var/zceesrvr; +
ln -s /var/shared/includes +
 /var/zceesrvr/includes; +
chown -R &USER:&GROUP $WLP_USER_DIR/servers/&SERVER
```

```
//*****
//* SET SYMBOLS
//*****
//EXPORT EXPORT SYMLIST=(*)
// SET JAVAHOME='/usr/lpp/java/J8.0_64'
// SET ZCEEPATH='/usr/lpp/IBM/zosconnect/v3r0'
// SET SERVER='openApi3'
// SET TEMPLATE='zosconnect:openApi3'
// SET WLPUSER='/var/ats/zosconnect'
// SET CONFIG='configDropins/overrides'
// SET USER='ATSSERV'
// SET GROUP='ATSGRP'
//*****
//** Step ZCEEAPI3 - Use the zosconnect command to create a server
//*****
//ZCEEAPI3 EXEC PGM=IKJEFT01,REGION=0M
//SYSTSPRT DD SYSOUT=*
//SYSERR DD SYSOUT=*
//STDOUT DD SYSOUT=*
//SYSTSIN DD *,SYMBOLS=EXEC SYS
BPXBATCH SH +
export JAVA_HOME=&JAVAHOME; +
export WLP_USER_DIR=&WLPUSER; +
&ZCEEPATH/bin/zosconnect create &SERVER +
--template=&TEMPLATE; +
ln -s $WLP_USER_DIR/servers/&SERVER /var/zceesrvr/&SERVER; +
ln -s /var/shared/includes/commonFeatures.xml +
 /var/zceesrvr/&CONFIG/commonFeatures.xml; +
ln -s /var/shared/includes/safSecurity.xml +
 /var/zceesrvr/&CONFIG/safSecurity.xml; +
cp /var/zceesrvr/properties/bootstrap.properties +
 /var/zceesrvr; +
cp /var/zceesrvr/properties/server.xml +
 /var/zceesrvr; +
ln -s /var/shared/includes +
 /var/zceesrvr/includes; +
chown -R &USER:&GROUP $WLP_USER_DIR/servers/&SERVER
```

RACF, Liberty and z/OS Connect Security Details and Options

z/OS Security – Range of options – Started Task IDs

On z/OS, the best practice for Liberty servers in production is that they run as ‘Started Tasks’ (STCs).



- Multiple servers
- All have same STC ID
- STC ID = File Owner ID

- Multiple servers
- All have same STC ID
- STC ID ≠ File Owner ID

- Multiple servers
- Different STC IDs
- STC IDs ≠ File Owner ID

Should all servers sharing WLP_USER_DIR share the same STC ID?
It is a matter of the degree of identity isolation that is required

z/OS Security: Assigning ID to started tasks: SAF STARTED class

The first question here is whether you wish to have a common started task ID that is shared among servers, or if you wish each server to have a unique ID

Then the second question is whether servers under a WLP_USER_DIR will share a common JCL start proc, or use unique start procs for each server

	<i>Common Identity per task</i>	<i>Unique Identities per task</i>
<i>Common JCL Procedure</i>	<pre>RDEFINE STARTED ZCEEPROC.* S ZCEEPROC,JOBNAM=server1,PARMS='server1' S ZCEEPROC,JOBNAM=server2,PARMS='server2'</pre>	<pre>RDEFINE STARTED ZCEEPROC.server1 RDEFINE STARTED ZCEEPROC.server2</pre>
<i>Unique JCL Procedure per server</i>	<pre>RDEFINE STARTED ZCEE*.* S ZCEESRV1,JOBNAM=server1,PARMS='server1' S ZCEESRV2,JOBNAM=server2,PARMS='server2'</pre>	<pre>RDEFINE STARTED ZCEESRV1.* RDEFINE STARTED ZCEESRV2.*</pre>

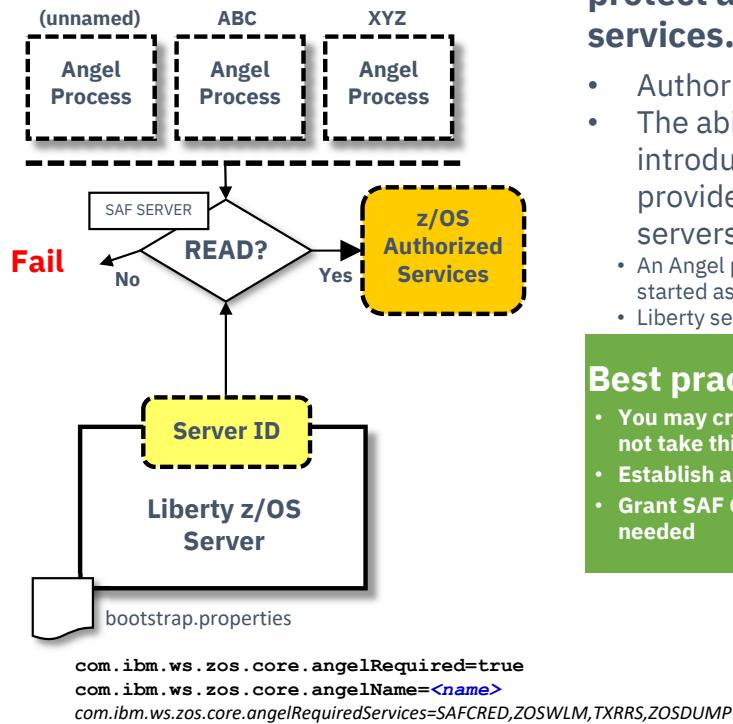
Note: Using unique JCL procedure eliminates the need to specify PARMS on the start commands

1. The same identity is used for all servers using a RACF STARTED class resource where the JCL procedure is discrete, and the job name is generic.
2. The same identity is used for all servers using a RACF STARTED class resource where both the JCL procedure and job names are generic
3. Different identities are used for each server using a RACF STARTED class resource where both the JCL procedure and job name are discrete.
4. Different identities are used for each server using a RACF STARTED class resource where the JCL procedure is discrete and job name is generic.

It's possible to use a combination of the above, even under the same WLP_USER_DIR. So there's no "one best answer" here. What's best is what's best for you.



z/OS Security: The Angel process – what is this about?



The Angel Process is a started task that is used to protect access to z/OS privileged or authorized services. This is done with SAF SERVER profiles.

- Authorized services include: WOLA, SAF, WLM, RRS, DUMP
- The ability to start multiple Angel processes on an LPAR was introduced in 16.0.0.4. This is called "Named Angels". It provides a way to separate Angel usage between Liberty servers:
 - An Angel process can be started with a NAME='<name>' parameter (or it can be started as a "default" without a name). The name may be up to 54 characters.
 - Liberty servers can be pointed at a specific Angel with a bootstrap property

Best practice:

- You may create separate named Angels for isolation of Test and Production, but do not take this practice too far. A few Angels, yes; dozens, no.
- Establish automation routines to start the Angels at IPL
- Grant SAF GROUP access to the SERVER profiles, then connect server IDs as needed

List of current Liberty Features

https://www.ibm.com/support/knowledgecenter/SSEQTP_liberty/com.ibm.websphere.wlp.doc/ae/rwlp_feat.html

z/OS Authorized service security: Angel Required Services

To use z/OS authorized services, you must have a Liberty Angel process and grant access for your Liberty server's SAF identity to use these services.

- LOCALCOM - Required to use *WebSphere Optimized Local Adapters* (WOLA).
- PRODMGR – Required to use IFAUSAGE services for SMF reporting.
- SAFCRED - Required to use SAF authorized user registry services and SAF authorization services.
- TXRRS - Required by the IBM® MQ resource adapter when the connection to IBM MQ is made in BINDINGS mode
- WOLA - Required to use *WebSphere Optimized Local Adapters* (WOLA).
- ZOSAIO - Required to use AsyncIO on z/OS.
- ZOSDUMP - Only required if asked to obtain an SVC dump by IBM service. It provides access to SVCDUMP services.
- ZOSWLM - Required to use WLM services. For more information, see [Measuring API workloads with WLM](#) at URL <https://www.ibm.com/docs/en/zos-connect/zosconnect/3.0?topic=considerations-measuring-api-workloads-wlm>

When a Liberty server connects to an angel process during server startup, it checks that the server's identity has access to the z/OS authorized services. By default, access checks are performed for all authorized services.

You can restrict the Liberty server to check and use only the authorized services it requires, which then makes other authorized services unavailable by using property, **com.ibm.ws.zos.core.angelRequiredServices**

The value for this property, **com.ibm.ws.zos.core.angelRequiredServices**, must be a comma-separated list of valid angel process services, as described above. This property must be specified with the **com.ibm.ws.zos.core.angelRequired** property set to **true**. Only these services, when properly specified, are the ones used by the server. **Lack of access to the angel process itself or any of these listed required services will cause a server startup failure.**



Use the *bootstrap.properties* file to required z/OS privileges

zceesrv1's bootstrap.properties

```
httpPort=9080
httpsPort=9443
ipicPort=1491
host=*
cicsHost=wg31.washington.ibm.com
network=ZOSCONN1
applid=ZOSCONN1
com.ibm.ws.zos.core.angelName=namedAngel
com.ibm.ws.zos.core.angelRequired=true
com.ibm.ws.zos.core.angelRequiredServices=SAFCRED,ZOSWLM,PRODMGR,ZOSAIO,TXRRS,LOCALCOM
```

zceesrv2's bootstrap.properties

```
httpPort=9090
httpsPort=9453
ipicPort=1492
host=wg31.washington.ibm.com
cicsHost=wg31.washington.ibm.com
network=ZOSCONN2
applid=ZOSCONN2
com.ibm.ws.zos.core.angelName=namedAngel
com.ibm.ws.zos.core.angelRequired=true
com.ibm.ws.zos.core.angelRequiredServices=SAFCRED,ZOSWLM,PRODMGR,ZOSAIO,TXRRS,LOCALCOM
```

z/OS Security: SAF SERVER profiles related to the Angel



Best practice:

- Establish all the SERVER profiles ahead of time. Existence of profile does not grant access; READ access does.
- Determine what access a server needs and grant only that; check "is available" messages in messages.log to verify

Tech/Tip: The SAFLOG parameter was added in a recent Liberty service. If this parameter is set to Y, additional security related messages will be written to the JES messages and console if a Liberty server does not have authorization to use an angel-controlled privileged function. See URL

https://www.ibm.com/support/knowledgecenter/SS7K4U_liberty/com.ibm.websphere.wlp.zseries.doc/ae/rwlp_newinrelease.html

Liberty 21.0.6 add a new property to identify required services, com.ibm.ws.zos.core.angelRequiredServices, for more details see URL

<https://www.ibm.com/docs/en/was-liberty/zos?topic=overview-process-types-zos>

Tech-Tip: SAF APPL and EJBRole Resources

Connect z/OS Connect users to a common group

CONNECT (FRED,USER1,JOHNSON) GROUP(ZCEEUSRS)

Define a APPL profile for the server's SAF profilePrefix and permit access

RDEFINE APPL BBGZDFLT UACC(NONE) OWNER(SYS1)

PERMIT BBGZDFLT CLASS(APPL) ACCESS(READ) ID(WSGUEST#, ZCEEUSRS)

SETROPTS RACLIST(APPL) REFRESH

Define an EJBROLE profile for the server's SAF profilePrefix and permit access

RDEFINE EJBROLE BBGZDFLT.zos.connect.access.roles.zosConnectAccess OWNER(SYS1) UACC(NONE)

PERMIT BBGZDFLT.zos.connect.access.roles.zosConnectAccess +

CLASS(EJBROLE) ID(ZCEEUSRS) ACCESS(READ)

Refresh the EJBROLE in storage profiles

SETROPTS RACLIST(EJBROLE) REFRESH

```
<safCredentials unauthenticatedUser="WSGUEST" profilePrefix="BBGZDFLT" />
```

- # https://www.ibm.com/support/knowledgecenter/SS7K4U/liberty/com.ibm.websphere.wlp.zseries.doc/ae/twlp_config_security_saf.html
<https://www.ibm.com/docs/en/zos-connect/zosconnect/3.0?topic=registry-saf-unauthenticated-user-id>

z/OS Connect Security

Overview

General security terms or considerations

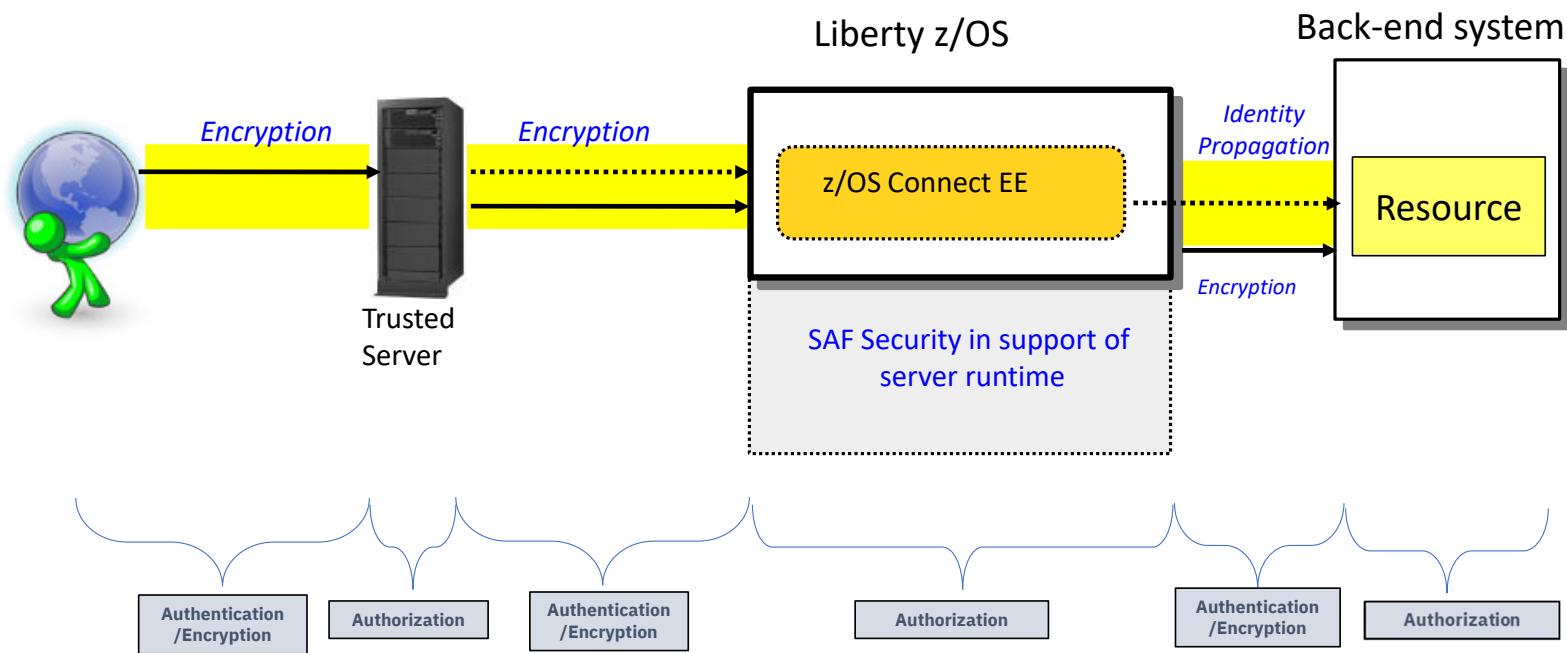
Security involves

- Identifying who or what is requesting access (**Authentication**)
 - Basic Authentication
 - Mutual Authentication using Transport Layer Security (TLS), formerly known as SSL
 - Third Party Tokens
- Ensuring that the message has not been altered in transit (**Data Integrity**) and ensuring the confidentiality of the message in transit (**Encryption**)
 - TLS (encrypting messages and using a digital signature)
- Controlling access (**Authorization**)
 - Is the authenticated identity authorized to access to z/OS Connect
 - Is the authenticated identity authorized to access a specific API, Services, etc.



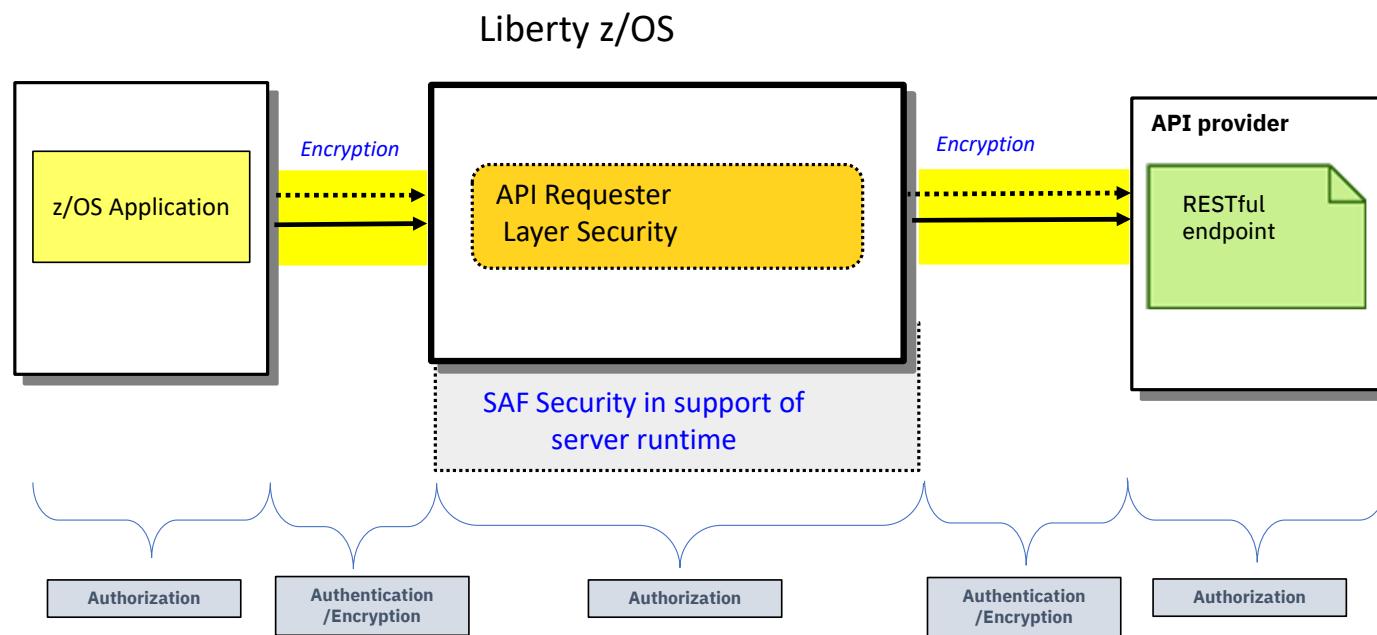


API Provider Authentication versus Authorization

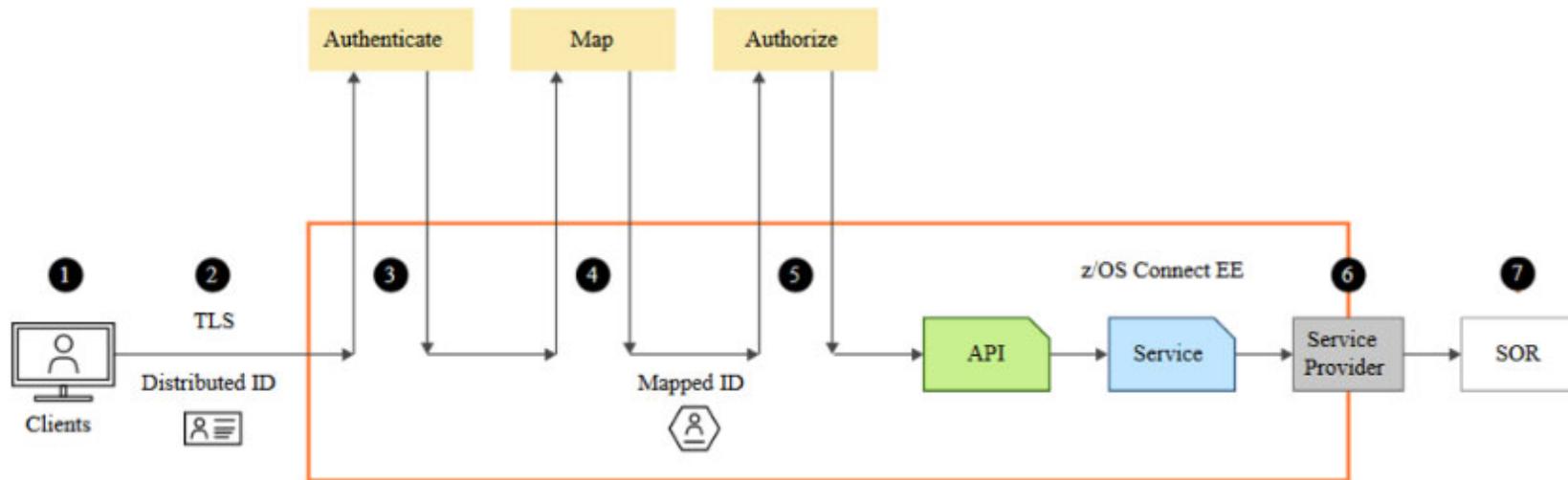




API Requester Authentication versus Authorization

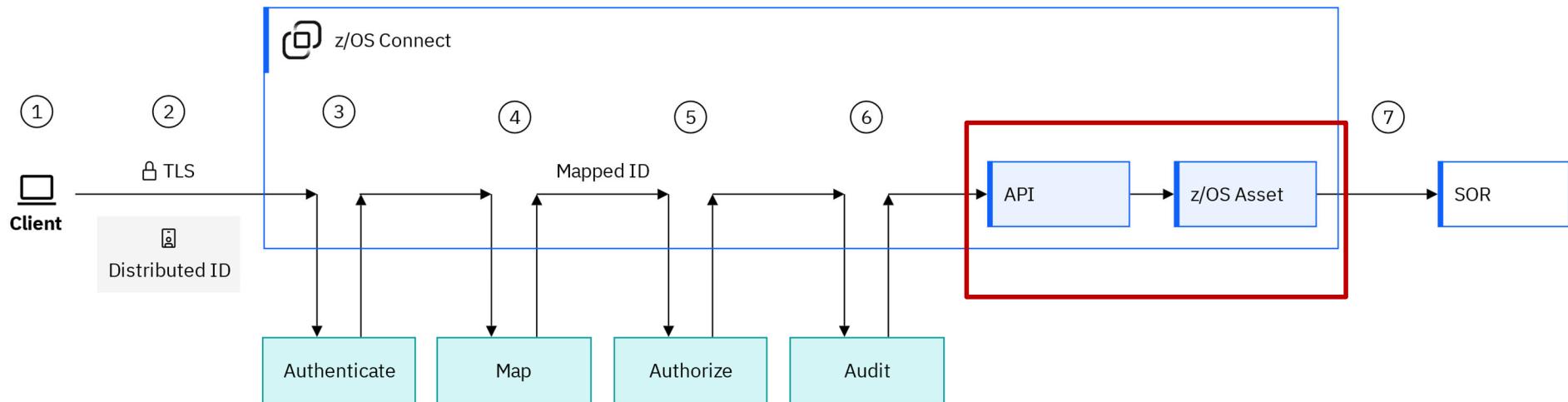


A typical z/OS Connect API Provider inbound security - inbound (OpenAPI 2)



1. The credentials provided by the client
2. Secure the connection to the Liberty server
3. Authenticate the client. This can be within the Liberty server or by requesting verification from a third-party server
4. Map the authenticated identity to a user ID in the user registry
5. Authorize the mapped user ID to connect to z/OS Connect EE and optionally authorize user to invoke actions on APIs
6. Secure the connection to the System of Record (SoR) and provide security credentials to be used to invoke the program or to access the data resource
7. The program or database request may run in the SoR under the mapped ID

Details of a typical z/OS Connect EE API Provider security flow - inbound (OpenAPI 3)



The flow includes the following security steps that can be performed by IBM z/OS Connect. The credentials are provided by the client. These can be a user ID and password, a JWT, or a TLS certificate.

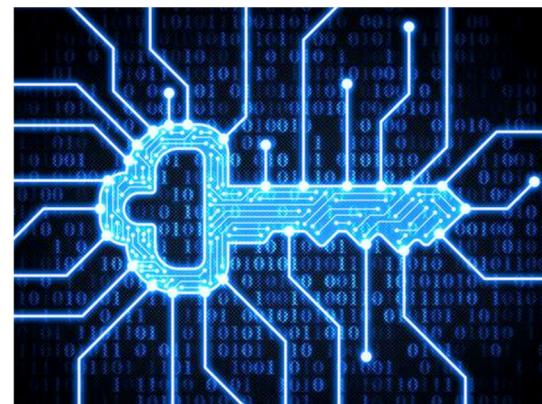
1. The credentials, including the identity, are passed on the connection between the client and IBM z/OS Connect. The identity is typically a distributed ID, such as an X.509 distinguished name and associated LDAP realm that originates from a remote system. Alternatively, the identity might be a SAF user ID. The data that is sent on the connection can be encrypted using TLS.
2. The client is authenticated. This can be within IBM z/OS Connect or by requesting verification from a third-party server.
3. The authenticated identity can be mapped to a user ID in the IBM z/OS Connect user registry.
4. The user is authorized to invoke the API operation if they have the required role.
5. The API request is audited by using the Liberty Audit feature.
6. The authenticated user identity can be propagated to the System of Record (SoR) when the SoR supports this capability. Alternatively, the SoR connection can be configured to use a functional identity.

**Now let's explore the security options for
inbound API Provider connections
and accessing z/OS resources**

General security terms or considerations

Security involves

- Identifying who or what is requesting access (**Authentication**)
 - Basic Authentication
 - Mutual Authentication using Transport Layer Security (TLS), formerly known as SSL
 - Third Party Tokens
- Ensuring that the message has not been altered in transit (**Data Integrity**) and ensuring the confidentiality of the message in transit (**Encryption**)
 - TLS (encrypting messages and using a digital signature)
- Controlling access (**Authorization**)
 - Is the authenticated identity authorized to access to z/OS Connect
 - Is the authenticated identity authorized to access a specific API, Services, etc.

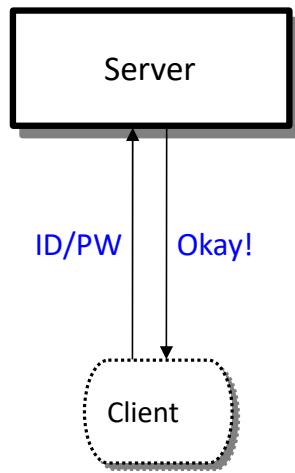




Liberty Authentication Options

Several different ways this can be accomplished:

Basic Authentication

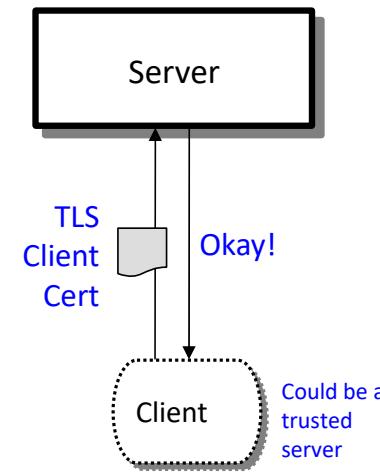


Client supplies ID/PW or ID/PassTicket

Server checks registry:

- Basic (server.xml)
- SAF

Client Certificate



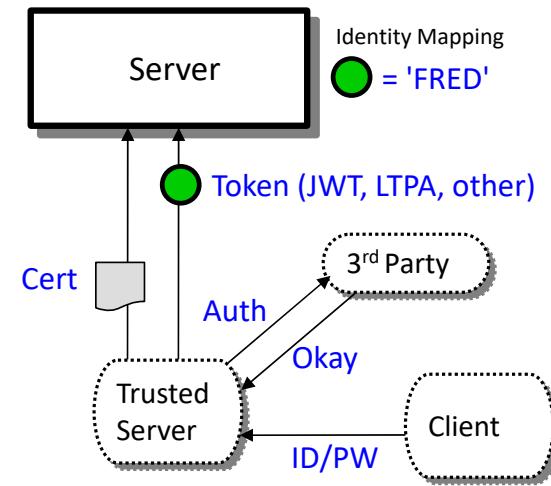
Client supplies client personal certificate

Server validates client personal certificate and maps it to an identity

Registry options:

- SAF

Third Party Authentication



Client authenticates to 3rd party sever

Client receives a trusted 3rd party token

Token flows to server and is mapped to an identity

Registry options:

- We may not need to know these details.



z/OS Connect Security server XML Authentication Configuration (OpenAPI 2)

- requireAuth - requires the client to provide credentials

```
<zosconnect_zosConnectManager  
    requireAuth="true|false"  
    requireSecure="true"/>  
  
<zosconnect_zosConnectAPIs>  
    <zosConnectAPI name="catalog"  
        requireAuth="true|false"  
        requireSecure="true"/>  
</zosconnect_zosConnectAPIs>  
  
<zosconnect_services>  
    <service id="selectByEmployee"  
        name="selectEmployee"  
        requireAuth="true|false"  
        requireSecure="true"/>  
</zosconnect_services>  
  
<zosconnect_apiRequesters>  
    requireAuth="true|false"  
    <apiRequester name="cscvincapi_1.0.0"  
        requireAuth="true|false"  
        requireSecure="true"/>  
</zosconnect_apiRequesters>
```

Globally, requires that users specify security credentials to be authenticated order to access APIs, services and API requesters, unless overridden on the specific resource definitions.

Requires that users specify security credentials to be authenticated in order to access the API.

Requires that users specify security credentials to be authenticated in order to directly access the service. This attribute is ignored when the service is invoked from an API, then only the API requireAuth attribute is relevant.

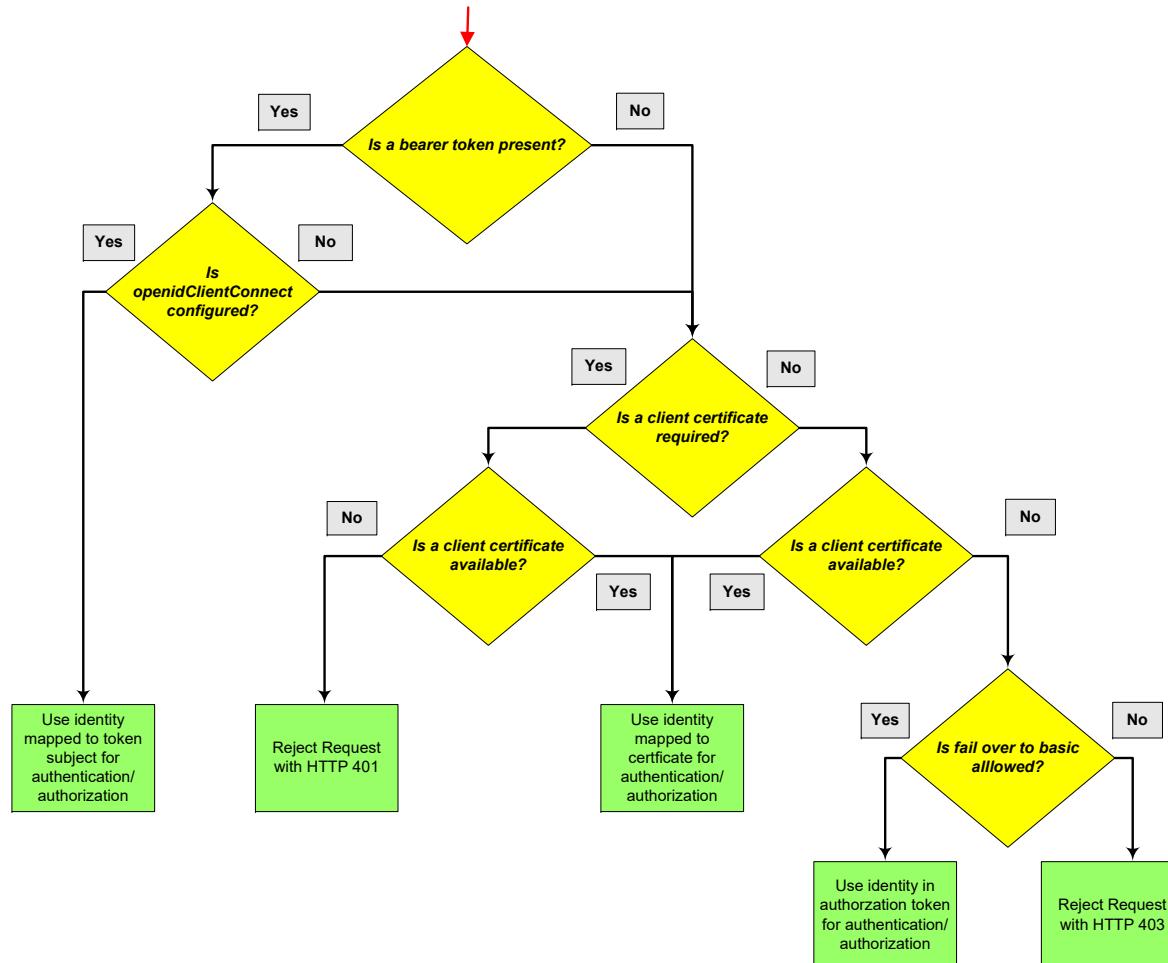
Requires that users specify security credentials to be authenticated in order to access all API requesters. If the requireAuth attribute is not set, the global setting on the zosconnect_zosConnectManager element is used instead, unless the requireAuth attribute is overridden on the specific API requester.

The requireAuth attribute controls whether an inbound request must provide credentials using one of the three authentication methods, e.g., basic, client certificate, or third-party token.

Note that there are no equivalent configuration elements for an OpenAPI 3 server.



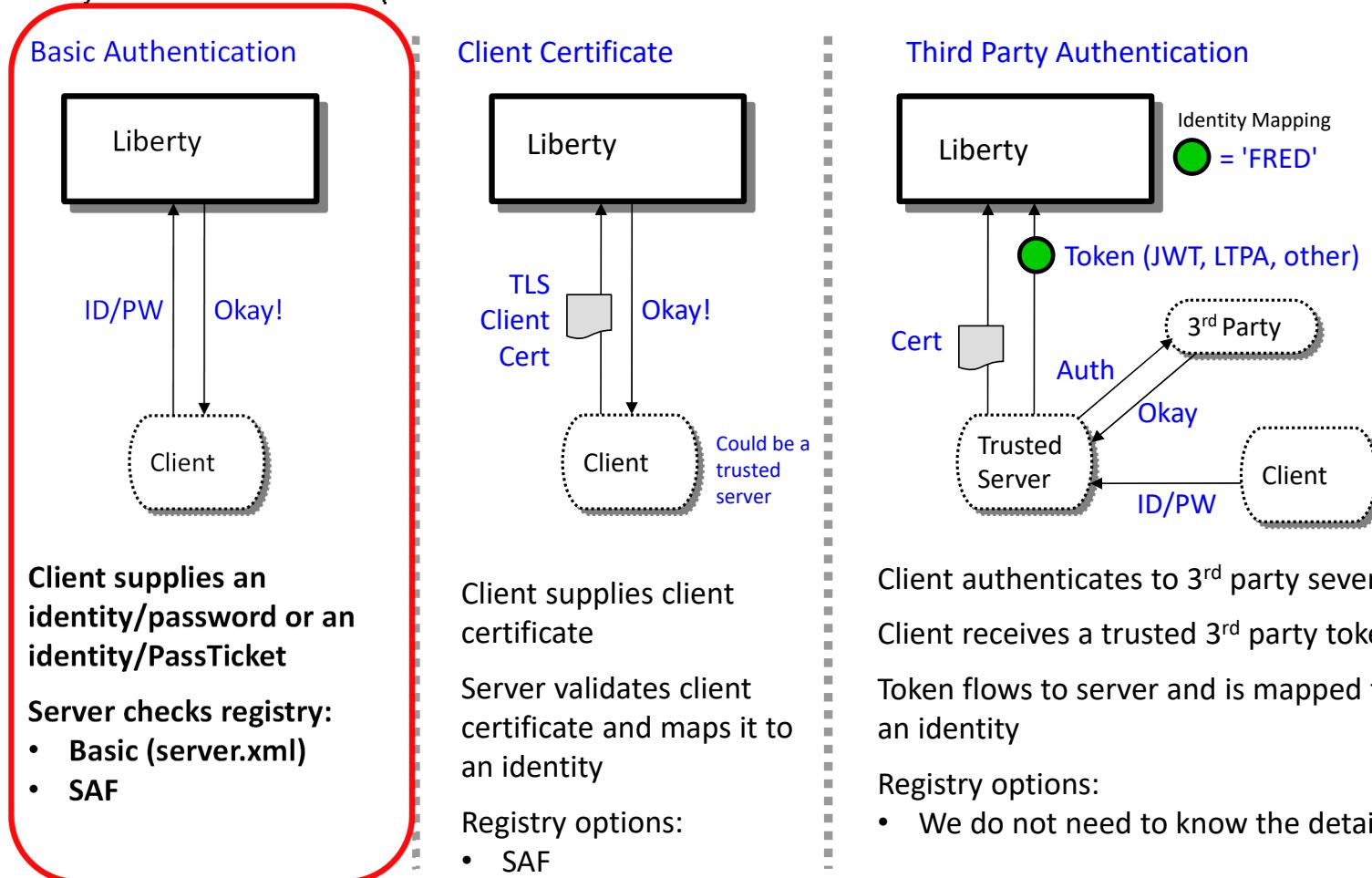
Authentication credential precedence order for determining authorization identity





Authentication - Basic Authentication

Several different ways this can be accomplished:





Basic authentication – Where the client provides an identity and password

- ❑ server XML security configuration:

```
<featureManager>
    <feature>appSecurity-2.0</feature>
    <feature>zosSecurity-1.0</feature>
</featureManager>

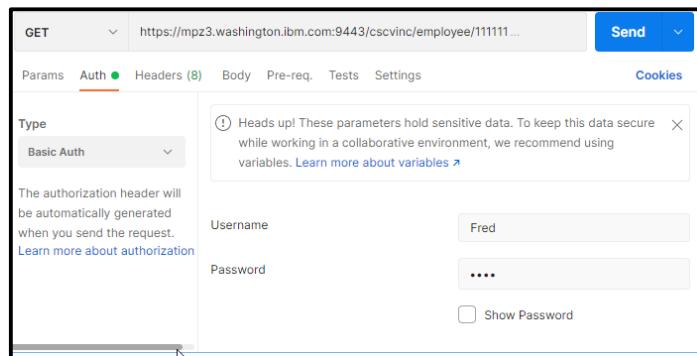
<webAppSecurity allowFailOverToBasicAuth="true" />

<safRegistry id="saf" />
<safAuthorization racRouteLog="ASIS" />
<safCredentials unauthenticatedUser="WSGUEST"
    profilePrefix="BBGZDFLT" />
```

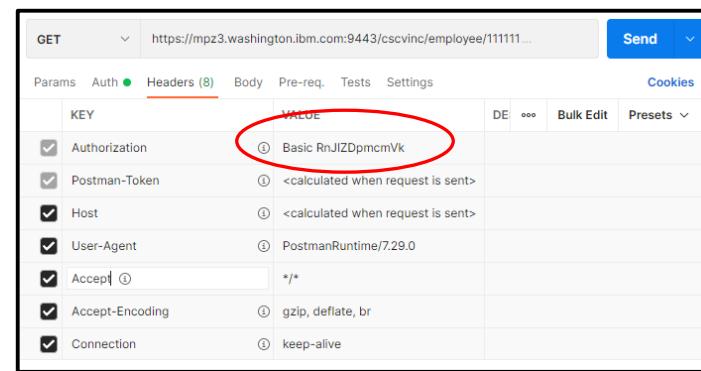
Note that these are Liberty configuration elements documented in the Liberty KC, i.e., no `zosconnect_` prefix.

- ❑ When sending a request to a Liberty server, basic authentication information (identity and password) is provided in the HTTP header in a *Basic Authorization* token with the identity and password encoded or formatted using Base64.

- An example with Postman:



The screenshot shows the Postman interface for a GET request to `https://mpz3.washington.ibm.com:9443/cscvinc/employee/111111...`. The 'Auth' tab is selected, and 'Basic Auth' is chosen from the dropdown. The 'Username' field contains 'Fred' and the 'Password' field contains '****'. A note in the sidebar says: 'The authorization header will be automatically generated when you send the request.' and 'Learn more about authorization'.



The screenshot shows the 'Headers' table in Postman. The 'Authorization' header is listed with the value `Basic RnJIZDpmcmVk`, which is circled in red. Other headers shown include Postman-Token, Host, User-Agent, Accept, Accept-Encoding, and Connection.

KEY	VALUE
Authorization	Basic RnJIZDpmcmVk
Postman-Token	<calculated when request is sent>
Host	<calculated when request is sent>
User-Agent	PostmanRuntime/7.29.0
Accept	*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive

There are multiple ways to provide an identity and password



- When sending a request to a Liberty server running z/OS Connect, basic authentication information (identity and password) is provided in the HTTP header in a *Basic Authorization* token with the identity and password encoded or formatted using Base64.
 - Examples using the API Explorer feature , cURL, and a Java client.

The screenshot shows the IBM Liberty REST API Documentation interface. A successful GET request to `/cscvinc/employee/000050` has been made, returning a response class of `Status 200 OK`. The response content type is set to `application/json`. The parameters `Authorization` and `employee` are highlighted with a red oval. A modal window titled `mpz3.washington.ibm.com:9443` is displayed, asking for sign-in with fields for `Username` (set to `user1`) and `Password` (set to `*****`). A `Sign In` button is visible in the bottom right of the modal.

```
Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

c:\z>curl -X GET --user FRED:SECRET https://mpz3.washington.ibm.com:9443/cscvinc/employee/111111
{"cscvincSelectServiceOperationResponse":{"cscvincContainer":{"response": {"CEIBRESP":0,"CEIBRESP2":0,"USERID":"CICSUSER","filea":{"employeeID": "111111","name": "C. BAKER","address": "OTTAWA, ONTARIO","phoneNumber": "51212003","date": "26 11 81","amount": "$0011.00"}}}}
c:\z>
```

```
49 URL url = new URL("https://wg31.washington.ibm.com:9453/db2/department?dept1=C01&dept2=C01");
50 System.out.println("URL: " + url);
51 HttpsURLConnection conn = (HttpsURLConnection) url.openConnection();
52 conn.setRequestMethod("GET");
53 conn.setRequestProperty("Content-Type", "application/json");
54 byte[] bytesEncoded = Base64.encodeBase64("fred@redpwd".getBytes());
55 conn.addRequestProperty("Authorization", new String(bytesEncoded));
56
57 try {
58
59     if (conn.getResponseCode() != 200) {
60         throw new RuntimeException("Failed : HTTP error code : " + conn.getResponseCode());
61     }
62     BufferedReader bufferReader = new BufferedReader(new InputStreamReader((conn.getInputStream())));
63     String output;
64     StringBuilder stringBuffer = new StringBuilder();
65     while ((output = bufferReader.readLine()) != null) {
66         stringBuffer.append(output);
67     }
68     JSONObject json = new JSONObject(stringBuffer.toString());
69     JSONArray jsonArray = json.getJSONArray("ResultSet 1 Output");
70     JSONObject jsonEntry = new JSONObject();
71     for (int index = 0; index < jsonArray.length(); index++) {
72         jsonEntry = jsonArray.getJSONObject(index);
73         if (jsonEntry.has("employeeNumber")){
```



Including a COBOL API Requester using basic authentication

- ❑ A MVS batch or IMS requester application sends basic authentication information (identity and password) by using environment variables.
 - BAQUSERNAME
 - BAQPASSWORD
- ❑ The environment variables can be provided in JCL using CEEOPTS DD statement:

```
//CEELOPTS DD *  
  POSIX(ON),  
  ENVAR("BAQURI=wg31.washington.ibm.com",  
"BAQPORT=9080",  
"BAQUSERNAME=USER1",  
"BAQPASSWORD=USER1")
```

Note that the z/OS Connect communications stub generates the Authentication header token we saw earlier

- ❑ Or, provided by using a CEEROPT or CEEUOPT module:

```
CEEROPT CSECT  
CEEROPT AMODE ANY  
CEEROPT RMODE ANY  
CEEXOPT POSIX=((ON),OVR),  
ENVAR=((BAQURI=wg31.washington.ibm.com',  
'BAQPORT=9120',  
'BAQUSERNAME=USER1',  
'BAQPASSWORD=USER1'),OVR),  
RPTOPTS=((ON),OVR)  
END
```

Tech/Tip: This is good opportunity to use a pass ticket rather than a password

Tech/Tip: A PassTicket provides an alternative to a password



- ❑ A PassTicket is generated by or for a client by using a secured sign-on key (whose value is masked or encrypted) to encrypt a valid *RACF identity* combined with the *application name* of the targeted resource. Also embedded in the PassTicket is a time stamp (based on the current Universal Coordinated Time (UCT)) which sets the time when the PassTicket will expire (usually 10 minutes).
- ❑ Access to PassTickets is managed using the RACF PTKTDATA class.
- ❑ For z/OS Connect, a RACF PassTicket can be used for basic authentication when connecting from any REST client on any platform to a z/OS Liberty server and for requests from a z/OS Connect server accessing IMS and Db2.
- ❑ ***PassTickets do not have to be generated on z/OS using RACF services.*** IBM has published the algorithm used to generate a PassTickets, see manual *z/OS Security Server RACF Macros and Interfaces, SA23-2288-40*. *Github has examples using Java, Python and other example are available on other sites.*

```
<safRegistry id="saf" />
  <safAuthorization racRouteLog="ASIS" />
  <safCredentials unauthenticatedUser="WSGUEST"
    profilePrefix="BBGZDFLT" />
```

Tech/Tip: Generating PassTickets on z/OS

- On z/OS, a COBOL user application can generate a pass tickets by calling RACF service IRRSPK00:

```
77 COMM-STUB-PGM-NAME          PIC X(8) VALUE 'BAQCSTUB'.
77 PTKT-STUB-PGM-NAME         PIC X(8) VALUE 'ATSPKTTC'.
*-----*
***** L I N K A G E   S E C T I O N *****
LINKAGE SECTION.
***** P R O C E D U R E S *****
PROCEDURE DIVISION using PARM-BUFFER.
*-----*
MAINLINE SECTION.
*-----*
* Common code *
*-----*
* initialize working storage variables
  INITIALIZE GET-REQUEST.
  INITIALIZE GET-RESPONSE.
CALL PTKT-STUB-PGM-NAME.
```

JOHNSON. PASSTCKT. SOURCE (ATSPKTTC)

```
*-----*
* Build IRRSPK00 parameters *
*-----*
      MOVE 0 to ws-length
      MOVE LENGTH OF identity to identity-length.
      INSPECT FUNCTION REVERSE (identity)
          TALLYING ws-length FOR ALL SPACES.
      SUBTRACT ws-length FROM identity-length.
      MOVE 0 to ws-length
      MOVE LENGTH OF applid to applid-length.
      INSPECT FUNCTION REVERSE (applid)
          TALLYING ws-length FOR ALL SPACES.
      SUBTRACT ws-length FROM applid-length.
      MOVE 8 to passTicket-length.
      MOVE 'NOTICKET' to passTicket.
      MOVE X'0003' to irr-functionCode.
      MOVE X'00000001' to irr-ticketOptions.
      SET irr-ticketOptions-ptr to ADDRESS OF irr-ticketOptions.
*-----*
* Call RACF service IRRSPK00 to obtain a pass ticket based *
*   on identity and applid *
*-----*
      PERFORM CALL-RACF.
      IF irr-safrc NOT = zero then
          DISPLAY "SAF_return_code:      " irr-safrc
          DISPLAY "RACF_return_code:     " irr-racfrc
          DISPLAY "RACF_reason_code:    " irr-racfrsn
      End-if
*-----*
* Call IRRSPK00 requesting a pass ticket *
*-----*
      CALL-RACF.
      CALL W-IRRSPK00 USING irr-workarea,
          IRR-ALET, irr-safrc,
          IRR-ALET, irr-racfrc,
          IRR-ALET, irr-racfrsn,
          IRR-ALET, irr-functionCode,
          irr-optionWord,
          IRR-PASSTICKET,
          irr-ticketOptions-ptr,
          IRR-IDENTITY,
          IRR-APPLID
```



Tech/Tip: RACF resources for using PassTickets

- ❑ First define a PTKTDATA resource using the *appName* assigned to the target subsystem:

```
RDEFINE PTKTDATA appName SSIGNON(KEYMASK(keymaskValue))
    APPLDATA('NO REPLAY PROTECTION')
```

Where:

appName is an application name assigned to the resource, e.g., BBGZDFLT
keymaskValue is the value of the secured sign-on application key, a 64-bit hex value
replayProtection indicates if a pass ticket can be reused

- ❑ Access to using PassTickets is controlled by another PTKTDATA resource, *IRRPTAUTH.appName.identity*. UPDATE access is required. For example, to use PassTickets to access a z/OS Connect server the resources below need to be defined and access permitted.

```
<safRegistry id="saf" />
  <safAuthorization racRouteLog="ASIS" />
  <safCredentials unauthenticatedUser="WSGUEST"
    profilePrefix="BBGZDFLT" />
```

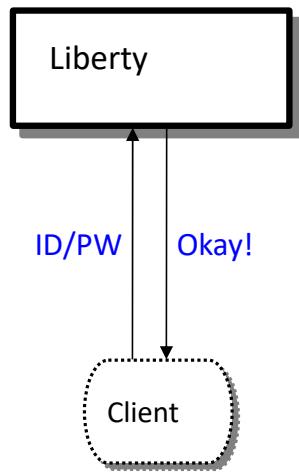
```
RDEFINE PTKTDATA BBGZDFLT SSIGNON(0123456789ABCDEF)
    APPLDATA('NO REPLAY PROTECTION') UACC(NONE)
RDEFINE PTKTDATA IRRPTAUTH.BBGZDFLT.* UACC(NONE)
PERMIT IRRPTAUTH.BBGZDFLT.* ID(LIBSERV) CLASS(PTKTDATA) ACCESS(UPDATE)
PERMIT IRRPTAUTH.BBGZDFLT.USER1 ID(USER1) CLASS(PTKTDATA) ACCESS(UPDATE)
```



Authentication - TLS Mutual Authentication

Several different ways this can be accomplished:

Basic Authentication



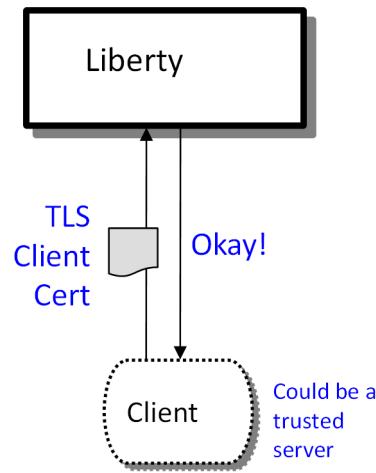
Server prompts for ID/PW

Client supplies ID/PW or ID/PassTicket

Server checks registry:

- Basic (server.xml)
- SAF

Client Certificate



Server prompts for client certificate.

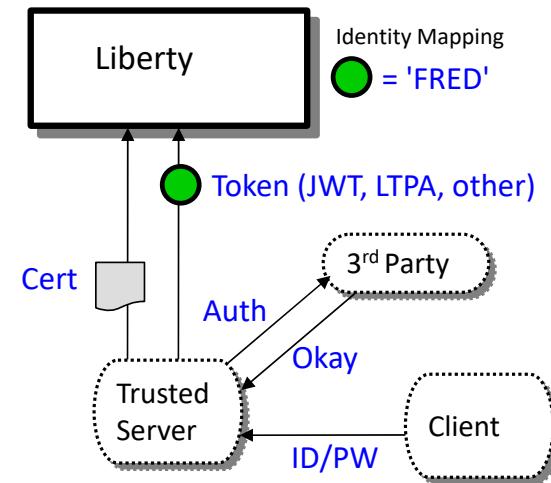
Client supplies personal certificate

Server validates client certificate and maps it to an identity

Registry options:

- SAF

Third Party Authentication



Client authenticates to 3rd party sever

Client receives a trusted 3rd party token

Token flows to Liberty z/OS and is mapped to an identity

Registry options:

- We may not need to know these details.

Liberty JSSE (HTTPS) server XML configuration



```
<!-- Enable features -->
<featureManager>
    <feature>transportSecurity-1.0</feature>
</featureManager>

<sslDefault sslRef="DefaultSSLSettings"
    outboundSSLRef="OutboundSSLSettings" />

<ssl id="DefaultSSLSettings"
    keyStoreRef="CellDefaultKeyStore"
    trustStoreRef="CellDefaultKeyStore"
    clientAuthenticationSupported="true"
    clientAuthentication="true"/>

<keyStore id="CellDefaultKeyStore"
    location="safkeyring:///Liberty.KeyRing"
    password="password" type="JCERACFKS"
    fileBased="false" readOnly="true" />

<ssl id="OutboundSSLSettings"
    keyStoreRef="OutboundKeyStore"
    trustStoreRef="OutboundKeyStore"/>

<keyStore id="OutboundKeyStore"
    location="safkeyring:///zCEE.KeyRing"
    password="password" type="JCERACFKS"
    fileBased="false" readOnly="true" />
```

SSL repertoires

Key ring for server certificate
send to for clients

Key ring for client connections to
server endpoints

OpenAPI 2

```
<zosconnect_zosConnectManager
    requireAuth="true"
    requireSecure="true|false"/>

<zosconnect_zosConnectAPIs>
    <zosConnectAPI name="catalog"
        requireAuth="true"
        requireSecure="true|false"/>
</zosconnect_zosConnectAPIs>

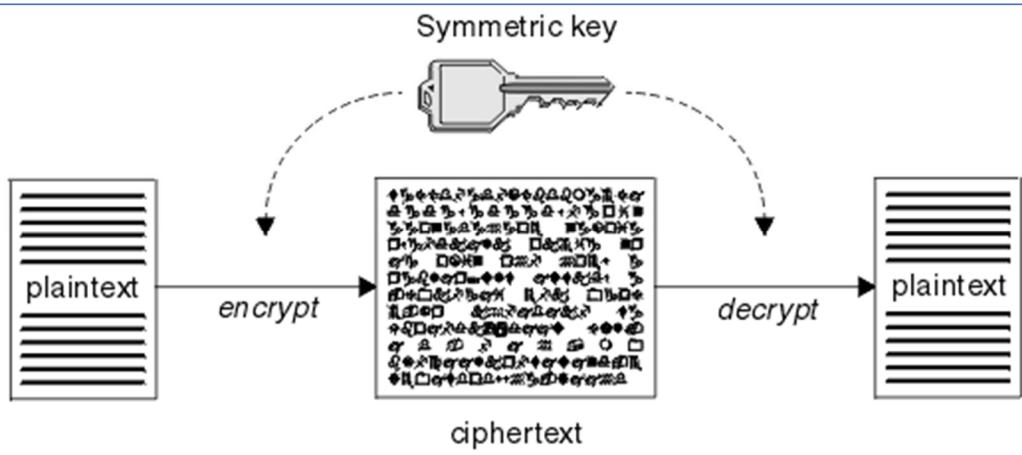
<zosconnect_services>
    <service id="selectByEmployee"
        name="selectEmployee"
        requireAuth="true"
        requireSecure="true|false"/>
</zosconnect_services>

<zosconnect_apiRequesters>
    requireAuth="true|false"
    <apiRequester name="cscvincapi_1.0.0"
        requireAuth="true"
        requireSecure="true|false"/>
</zosconnect_apiRequesters>
```

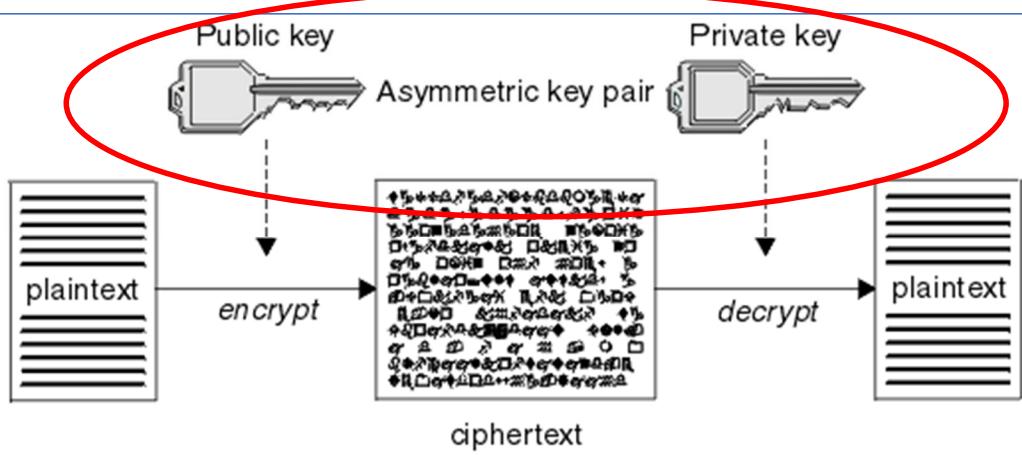
safkeyring:///KeyRing v safkeyring://owner/KeyRing

Tech/Tip: Regarding *clientAuthentication* and *clientAuthenticationSupported*. Understand the implications of the interactions between these attributes. There may instances where you want to use HTTPS, but not always with mutual authentication Consider setting *clientAuthentication* to false when setting *clientAuthenticationSupported* to true.

Tech-Tip: Symmetric key v. Asymmetric key pairs



A symmetric key is a key shared by the endpoints. Both endpoints use the same key to encrypt and decrypt messages.



An asymmetric key pair is the preferred solution. There is no risk of compromise by sending a symmetric or shared key outside of a protected communication flow.

A message encrypted with a public key can only be decrypted by endpoint that has the private key. The privacy of the messages is ensured.

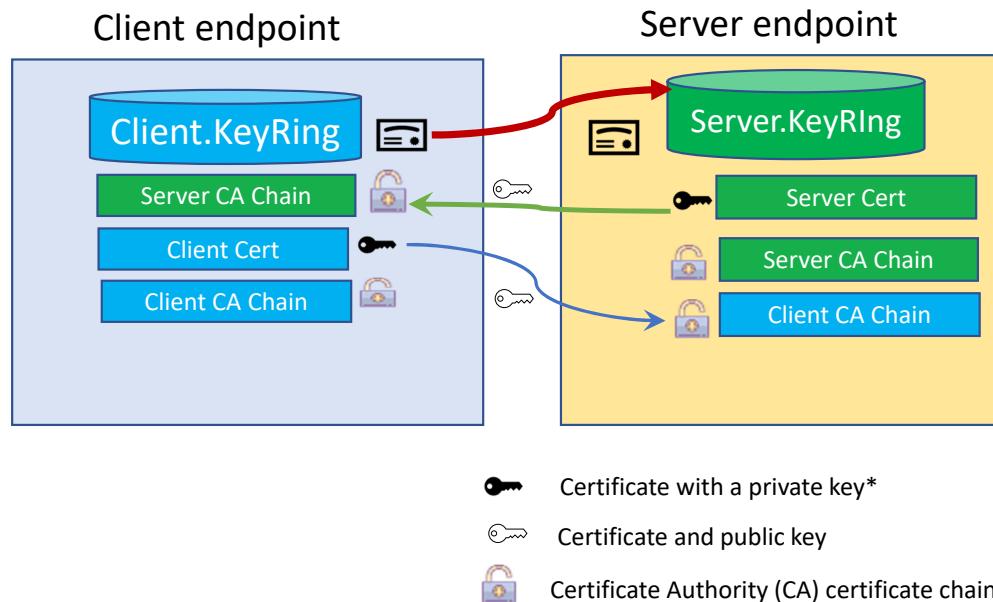
If an endpoint can successfully decrypt a message message encrypted received with a private key, the endpoint sending the message has successfully asserted its validity by proving it has the private key used to encrypt the message.

The basic TLS Handshake Flow (HTTPS)

The HTTPS protocol involves a TLS handshake –

Server Authentication (always occurs when HTTPS is the protocol)

Mutual Authentication (optional, at the request of the server endpoint)



*For server and/or mutual authentication to work, the endpoint sending the client certificate must use a personal certificate with a private key. The private key is required to decrypt (or encrypt) a message digest that is sent from the other endpoint during the handshake flow. Generation of a message digest also requires access to the CA certificate used to sign the certificate.

#Refers to the set or of certificates used to issue the server or client personal certificate including any intermediate certificates up to and including the root CA.

Tech-Tip: Key rings(non-virtual) and accessing a certificate's private keys

Two types of RACF profiles resources are used to control access to key ring and certificates

- **RDTALIB** for controlling access to a specific key ring
- **FACILITY** for controlling access to key rings globally

User certificates (connected to the key ring with usage PERSONAL)

- Global profiles uses the **FACILITY** resource **IRR.DIGTCERT.LISTRING**:
 - **READ** access is required to access one's own key ring and private key
 - **UPDATE** access is required to access another user's key private key
- Specific key rings uses the **RDTALIB** class **<ring owner>.<ring name>.LST**
 - **READ** access is required to access one's own private key
 - **UPDATE** access is required to access another identity's private keys

CERTAUTH and SITE certificates (connected to the key ring with usage PERSONAL)

- Global profiles uses the **FACILITY** resource **IRR.DIGTCERT.GENCERT**:
 - **CONTROL** access is required to access a CERTAUTH or SITE certificate private key ring
- Specific key rings uses the **RDTALIB** class **<ring owner>.<ring name>.LST**
 - **CONTROL** access is required to access the private keys of CERTAUTH and SITE certificates

Remember: When switching from global FACILITY class profiles to specific ring RDTALIB class profiles, the RDTALIB resources will be checked first

Tech/Tip: Details of the flow with mutual authentication (TLS 1.2)

1. A Client sends a request to server for a protected session in a ***ClientHello*** message. Included in the request is the TLS capabilities of the client (e.g., TLS 1.2 or 1.3) and a list of supported ciphers in preference order.
2. The server selects the TLS version and selects cipher from the list sent by the client and returns this information in a ***ServerHello*** message.
3. The server's certificate public information (including the **public key**) is sent to the client in a ***Certificate*** message.
4. The server sends cryptographic information for the client to use for encrypting a pre-master key in a ***Server key exchange*** message.
- 5. For mutual authentication, the server sends a *CertificateRequest* message requesting a client's personal certificate.**
6. The server concludes by sending a ***ServerHelloDone*** message.
7. The client verifies the server's certificate with its trust store.
- 8. If mutual authentication is requested, the client sends its public personal certificate information in a *Certificate* message**
9. The client then uses the ***server's public key*** to generate and encrypt a 48 byte "premaster secret" message which is sent to the server in a ***ClientKeyExchange*** message.
- 10. When mutual authentication is requested, a digitally signature (hashed) of the concatenation of all previous handshake messages is encrypted with the client's private key sent in a *CertificateVerify* message.**
11. The ***Change Cipher*** message is used to change the cipher used during the handshake so all subsequent messages will be encrypted using a different cipher.
12. The server uses its ***private key*** to decrypt the "premaster secret" message (**only the private key can be used to decrypt the message**).
- 13. If mutual authentication is requested, the server verifies the client's personal certificate with its key ring and uses the client's public key to decrypt and verify the message sent in the *CertificateVerify* message.**
14. Both the Client and Server use the "premaster secret" to compute a 'master secret', also known as "shared secret" or "session key" (symmetric encryption)
15. Client and server will use this "shared secret" or "session key" to encrypt messages sent between the endpoints.

Tech/Tip: A note on cipher suite names

A CipherSuite is a suite of cryptographic algorithms used by a TLS connection. A suite comprises three distinct algorithms:

- The key exchange and authentication algorithm, used during the handshake
- The encryption algorithm, used to encipher the data
- The MAC (Message Authentication Code) algorithm, used to generate the message digest

There are several options for each component of the suite, but only certain combinations are valid when specified for a TLS connection. The name of a valid CipherSuite defines the combination of algorithms used. For example, the CipherSuite ***TLS_RSA_WITH_AES_128_CBC_SHA*** specifies:

- The RSA key exchange and authentication algorithm
- The AES encryption algorithm, using a 128-bit key and cipher block chaining (CBC) mode
- The SHA-1 Message Authentication Code (MAC)

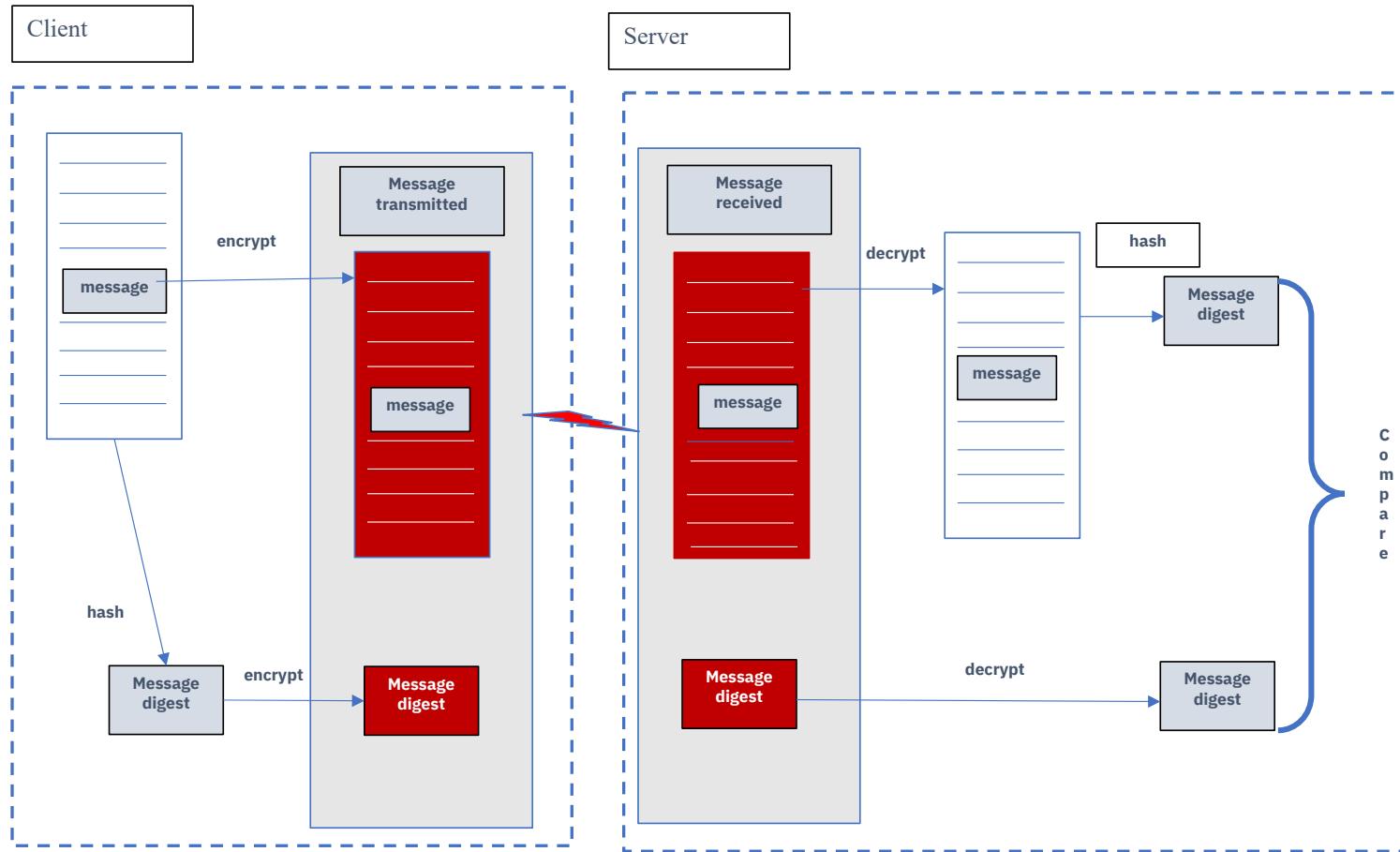
Note					
To use some CipherSuites, the 'unrestricted' policy files need to be configured in the JRE. For more details of how policy files are set up in an SDK or JRE, see the IBM SDK Policy files topic in the Security Reference for IBM SDK, Java Technology Edition for the version you are using.					
Table 1. CipherSpecs supported by IBM MQ and their equivalent CipherSuites					
CipherSpec	Equivalent CipherSuite (IBM JRE)	Equivalent CipherSuite (Oracle JRE)	Protocol	FIPS 140-2 compatible	
ECDHE_ECDSA_3DES_EDE_CBC_SHA256	SSL_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA	TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA	TLS 1.2	yes	
ECDHE_ECDSA_AES_128_CBC_SHA256	SSL_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	TLS 1.2	yes	
ECDHE_ECDSA_AES_128_GCM_SHA256	SSL_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	TLS 1.2	yes	
ECDHE_ECDSA_AES_256_CBC_SHA384	SSL_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	TLS 1.2	yes	
ECDHE_ECDSA_AES_256_GCM_SHA384	SSL_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	TLS 1.2	yes	
ECDHE_ECDSA_NULL_SHA256	SSL_ECDHE_ECDSA_WITH_NULL_SHA	TLS_ECDHE_ECDSA_WITH_NULL_SHA	TLS 1.2	no	
ECDHE_ECDSA_RC4_128_SHA256	SSL_ECDHE_ECDSA_WITH_RC4_128_SHA	TLS_ECDHE_ECDSA_WITH_RC4_128_SHA	TLS 1.2	no	
ECDHE_RSA_3DES_EDE_CBC_SHA256	SSL_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA	TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA	TLS 1.2	yes	
ECDHE_RSA_AES_128_CBC_SHA256	SSL_ECDHE_RSA_WITH_AES_128_CBC_SHA256	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	TLS 1.2	yes	
ECDHE_RSA_AES_128_GCM_SHA256	SSL_ECDHE_RSA_WITH_AES_128_GCM_SHA256	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	TLS 1.2	yes	
ECDHE_RSA_AES_256_CBC_SHA384	SSL_ECDHE_RSA_WITH_AES_256_CBC_SHA384	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	TLS 1.2	yes	
ECDHE_RSA_AES_256_GCM_SHA384	SSL_ECDHE_RSA_WITH_AES_256_GCM_SHA384	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	TLS 1.2	yes	
ECDHE_RSA_NULL_SHA256	SSL_ECDHE_RSA_WITH_NULL_SHA	TLS_ECDHE_RSA_WITH_NULL_SHA	TLS 1.2	no	
ECDHE_RSA_RC4_128_SHA256	SSL_ECDHE_RSA_WITH_RC4_128_SHA	TLS_ECDHE_RSA_WITH_RC4_128_SHA	TLS 1.2	no	

https://www.ibm.com/support/knowledgecenter/SSFKSJ_9.1.0/com.ibm.mq.dev.doc/q113210_.htm

mitchj@us.ibm.com

© 2017, 2022 IBM Corporation
Slide 74

Message Integrity and Encryption (client to server endpoint)



Tech/Tip: RACF digital certificate (RACDCERT) command review

```
RACDCERT ID(LIBSERV) GENCERT SUBJECTSDN(CN('wg31.washington.ibm.com') +  
O('IBM') OU('LIBERTY')) WITHLABEL('Liberty Server Cert') ALTNAMES(DOMAIN('wg31z.washington.ibm.com'))  
RACDCERT ID(LIBSERV) GENREQ(LABEL('Liberty Server Cert')) DSN(CERT.REQ)
```

Send the certificate to your Certificate Authority to be signed

```
racdcert CERTAUTH withlabel('Liberty CA') add('USER1.LIBCA.PEM') TRUST  
racdcert id(LIBSERV) withlabel('Liberty Server Cert') add('LIBSERV.P12') password('secret') TRUST
```

```
/* Create Liberty key ring and connect CA and personal certificates */  
racdcert id(libserv) addring(Liberty.KeyRing)  
racdcert id(libserv) connect(ring(Liberty.KeyRing) label('CICS CA') certauth usage(certauth))  
racdcert id(libserv) connect(ring(Liberty.KeyRing) label('Liberty CA') certauth usage(certauth))  
/* Connect default personal certificate */  
racdcert id(libserv) connect(ring(Liberty.KeyRing) label('Liberty Client Cert') default
```

```
setropts raclist(digtcert) refresh
```

Broadcom Support web pages

Site of *What ACF2 security setup is needed for IBM's z/OS Connect Enterprise Edition V3.0?*

<https://knowledge.broadcom.com/external/article/128597/what-acf2-security-setup-is-needed-for-i.html>

Site of *ACF2 setup for z/OS Connect Enterprise Edition V3.0*

<https://knowledge.broadcom.com/external/article/142172/acf2-setup-for-zos-connect-enterprise-ed.html>

Site of *Setting up Liberty Server for z/OS with Top Secret*

<https://knowledge.broadcom.com/external/article/37272/setting-up-liberty-server-for-zos-with-t.html>

Tech/Tip: RACF Certificate Filtering and Mapping

Filters for mapping certificates can be created with a RACDCERT command.

- Enter command RACDCERT ID MAP to create a filter that assigns RACF identity ATSUSER to any digital certificate signed with the ATS client signer certificate and where the subject is organizational unit ATS in organization IBM.

```
racdcert id(atsuser) map sdnfilter('OU=ATS.O=IBM')
idnfilter('CN=ATS Client CA.OU=ATS.O=IBM') withlabel('ATS USERS')
```

- Enter command RACDCERT ID MAP to create a filter that assigns RACF identity OTHUSER to any digital certificate signed by the ATS client signer certificate and where the subject is in organization IBM.

```
racdcert id(othuser) map sdnfilter('O=IBM')
idnfilter('O=IBM') withlabel('IBM USERS')
```

- Refresh the in-storage profiles for digital certificate maps.

```
SETRPTS RACLIST(DIGTNMAP) REFRESH
```

The Liberty JSSE server XML configuration for outbound connections



```
<!-- Enable features -->
<featureManager>
    <feature>transportSecurity-1.0</feature>
</featureManager>

<ssl id="cicsTLSSettings"
    keyStoreRef="CICSKeyStore"
    trustStoreRef="CICSKeyStore"
    clientKeyAlias="Liberty Client Cert"/>
<keyStore id="CICSKeyStore"
    location="safkeyring:///Liberty.CICS.KeyRing"
    password="password" type="JCERACFKS"
    fileBased="false" readOnly="true" />
<ssl id="db2TLSSettings"
    keyStoreRef="Db2KeyStore"
    trustStoreRef="Db2KeyStore"
    clientKeyAlias="Liberty Client Cert"/>
<keyStore id="Db2KeyStore"
    location="safkeyring:///Liberty.Db2.KeyRing"
    password="password" type="JCERACFKS"
    fileBased="false" readOnly="true" />
<ssl id="otherTLSSettings"
    keyStoreRef="OtherKeyStore"
    trustStoreRef="OtherKeyStore">
    <outboundConnection
        host="wg31.washington.ibm.com"
        port="9555"
        clientCertificate="Client Cert"/>
</ssl>
<keyStore id="OtherKeyStore"
    location="safkeyring:///Other.KeyRing"
    password="password" type="JCERACFKS"
    fileBased="false" readOnly="true" />
```

```
<zosconnect_authorizationServer sslCertsRef="SSL repertoire"/>
<zosconnect_cicsIpicConnection sslCertsRef="cicsTLSSettings"/>
<zosconnect_db2Connection sslCertsRef="db2TLSSettings"> *
<zosconnect_endpointConnect sslCertsRef= "SSL repertoire"/>
<zosconnect_zosConnectRestClient sslCertsRef="SSL repertoire"/>
<zosconnect_zosConnectServiceRestClientConnection sslCertsRef="SSL repertoire"/>
```

F BAQSTRT,REFRESH,KEYSTORE
F BAQSTRT,REFRESH,KEYSTORE,ID=CICSKeyStore
F BAQSTRT,REFRESH,KEYSTORE,ID=Db2KeyStore
F BAQSTRT,REFRESH,KEYSTORE,ID=OtherKeyStore



Tech/Tip: Combining TLS mutual and basic authentication

```
*****  
/* SET SYMBOLS  
*****  
EXPORT EXPORT SYMLIST=(*  
SET CURL= '/usr/lpp/rocket/curl'  
*****  
/* CURL Procedure  
*****  
CURL PROC  
CURL EXEC PGM=IKJEFT01,REGION=0M  
SYSTSPRT DD SYSOUT=*  
SYSERR DD SYSOUT=*  
STDOUT DD SYSOUT=*  
PEND  
*****  
/* STEP CURL - use curl to deploy API cscvinc  
*****  
DEPLOY EXEC CURL  
BPXBATCH SH export CURL=&CURL; +  
$CURL/bin/curl -X PUT -s +  
--cacert /u/johnson/CERTAUTH.PEM --user FRED:FRED +  
https://wg31.washington.ibm.com:9445/zosConnect/apis/cscvinc?status=sto+  
pped > null; +  
$CURL/bin/curl -X DELETE -s +  
--cacert /u/johnson/CERTAUTH.PEM --user FRED:FRED +  
https://wg31.washington.ibm.com:9445/zosConnect/apis/cscvinc > null; +  
$CURL/bin/curl -X POST -s +  
--cacert /u/johnson/CERTAUTH.PEM --user FRED:FRED +  
--data-binary @/u/johnson/cscvinc.aar +  
--header "Content-Type: application/zip" +  
https://wg31.washington.ibm.com:9445/zosConnect/apis  
*****  
/* STEP CURL - use curl to invoke the API cscvinc  
*****  
INVOKE EXEC CURL  
SYSTSIN DD *,SYMBOLS=EXECSYS  
BPXBATCH SH export CURL=&CURL; $CURL/bin/curl -X GET -s +  
--cacert /u/johnson/CERTAUTH.PEM --user FRED:FRED +  
https://wg31.washington.ibm.com:9445/cscvinc/employee/000100
```

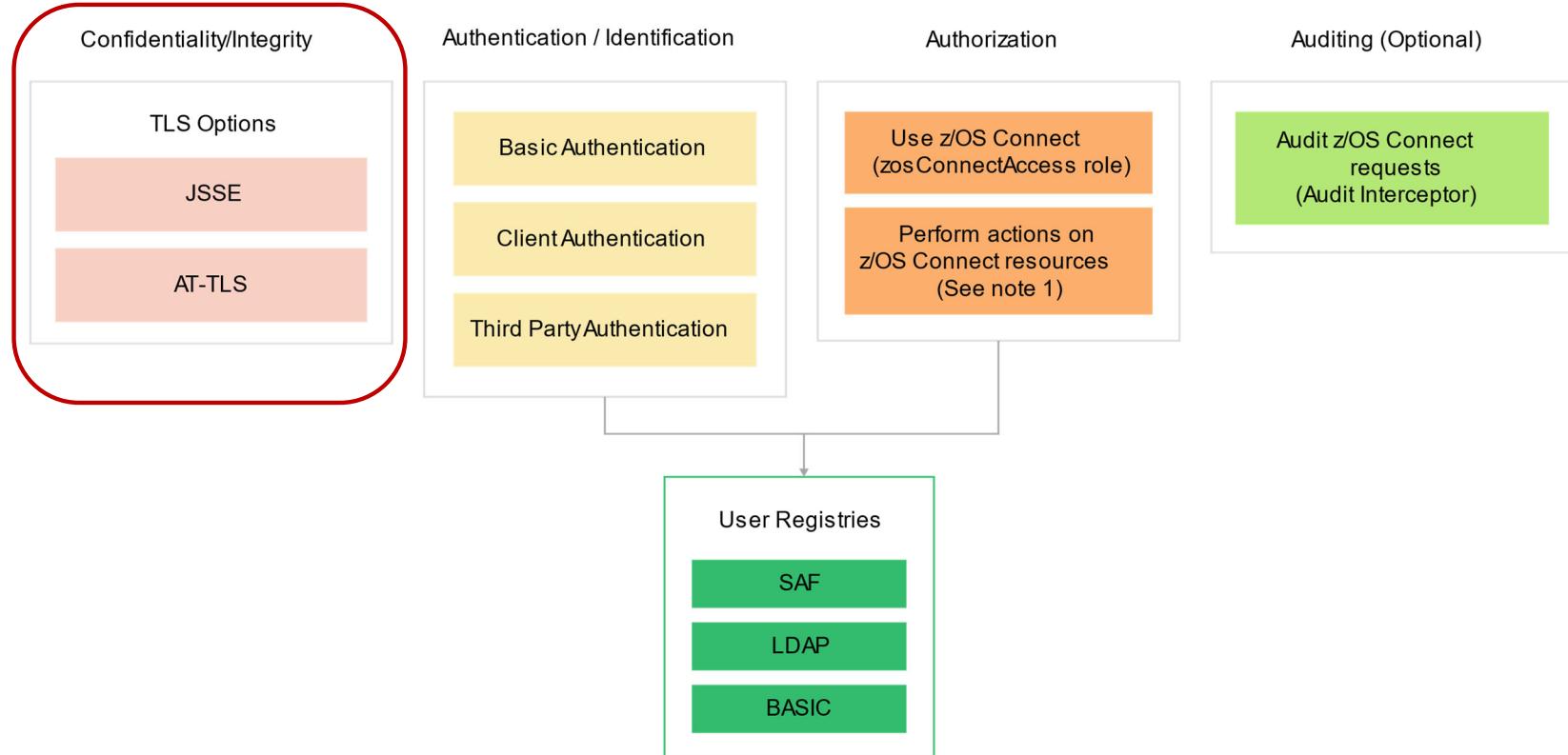
```
<httpEndpoint id="defaultHttpEndpoint"  
host="*"  
httpPort="9080"  
httpsPort="9443" />  
  
<sslDefault sslRef="DefaultSSLSettings"  
outboundSSLRef="DefaultSSLSettings" />  
  
<ssl id="DefaultSSLSettings"  
keyStoreRef="CellDefaultKeyStore"  
trustStoreRef="CellDefaultKeyStore"  
clientAuthenticationSupported="true"  
clientAuthentication="true"/>  
  
<keyStore id="CellDefaultKeyStore"  
location="safkeyring:///Liberty.KeyRing"  
password="password" type="JCERACFKS"  
fileBased="false" readOnly="true" />
```

```
<httpEndpoint id="AdminHttpEndpoint"  
host="*"  
httpPort="-1"  
httpsPort="9445"  
sslOptionsRef="mySSLOptions"/>  
  
<ssLOptions id="mySSLOptions"  
sslRef="BatchSSLSettings" />  
  
<ssl id="BatchSSLSettings"  
keyStoreRef="CellDefaultKeyStore"  
trustStoreRef="CellDefaultKeyStore"  
clientAuthenticationSupported="true"  
clientAuthentication="false"/>
```

<https://www.rocketsoftware.com/platforms/ibm-z/curl-for-zos>



Liberty and z/OS Connect EE security options (OpenAPI 2)

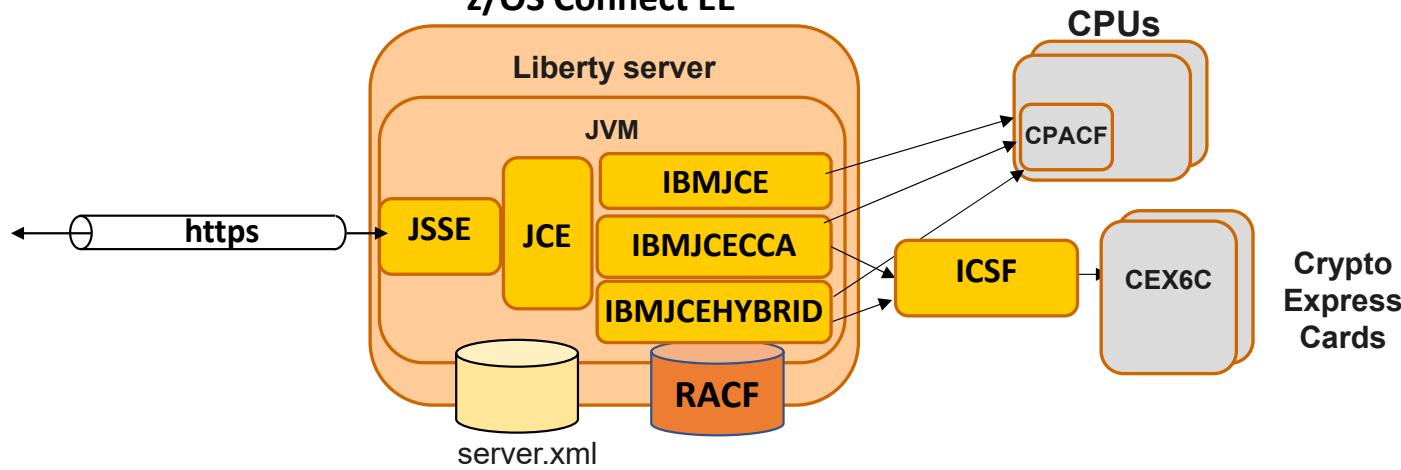


The actions which can be controlled by authorization are deploying, querying, updating, starting, stopping and deleting of APIs, services and API requesters.

Using JSSE with Liberty



The server XML configuration defines the **HTTPS ports, key rings, and other JSSE attributes**
z/OS Connect EE

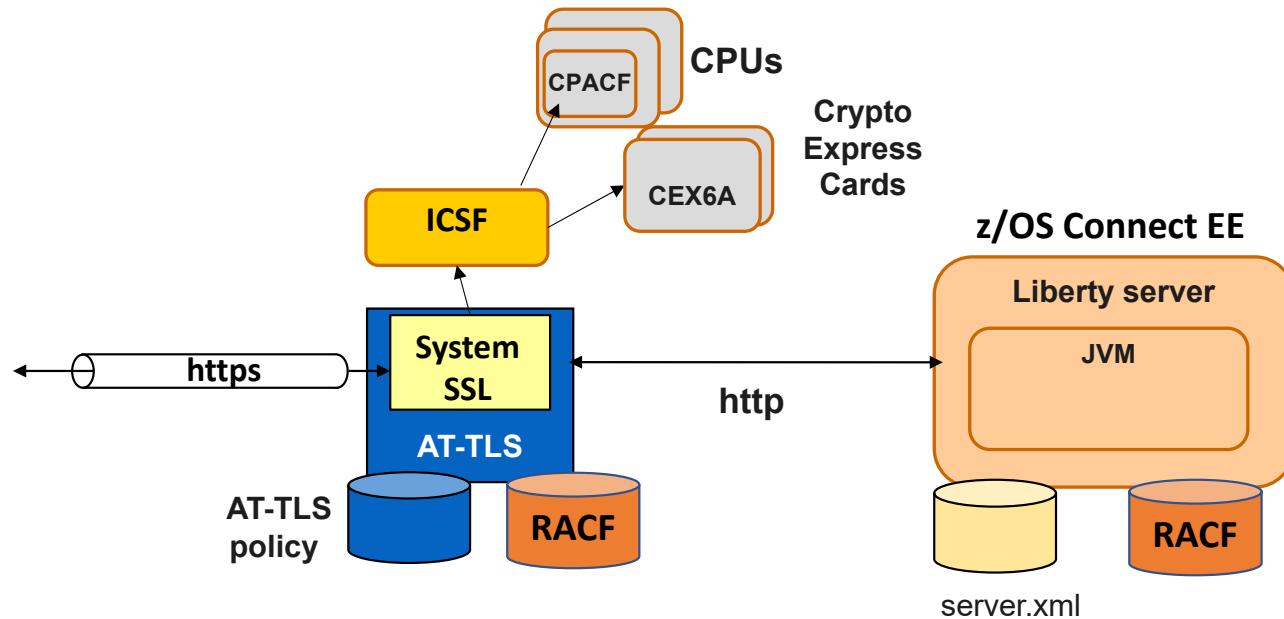


- z/OS Connect EE support for TLS is based on **Liberty** server support
- **Java Secure Socket Extension (JSSE)** API provides framework and Java implementation of TLS protocols used by Liberty HTTPS support
- **Java Cryptography Extension (JCE)** is standard extension to the Java Platform that provides implementation for cryptographic services
- **IBM Java SDK for z/OS** provides three different JCE providers, **IBMJCE**, **IBMJCECCA** and **IBMJCEHYBRID**.
- The JCE providers access **CPACF (CP Assist for Cryptographic Functions)** directly, therefore keep your Java service levels current.



Using AT-TLS with Liberty

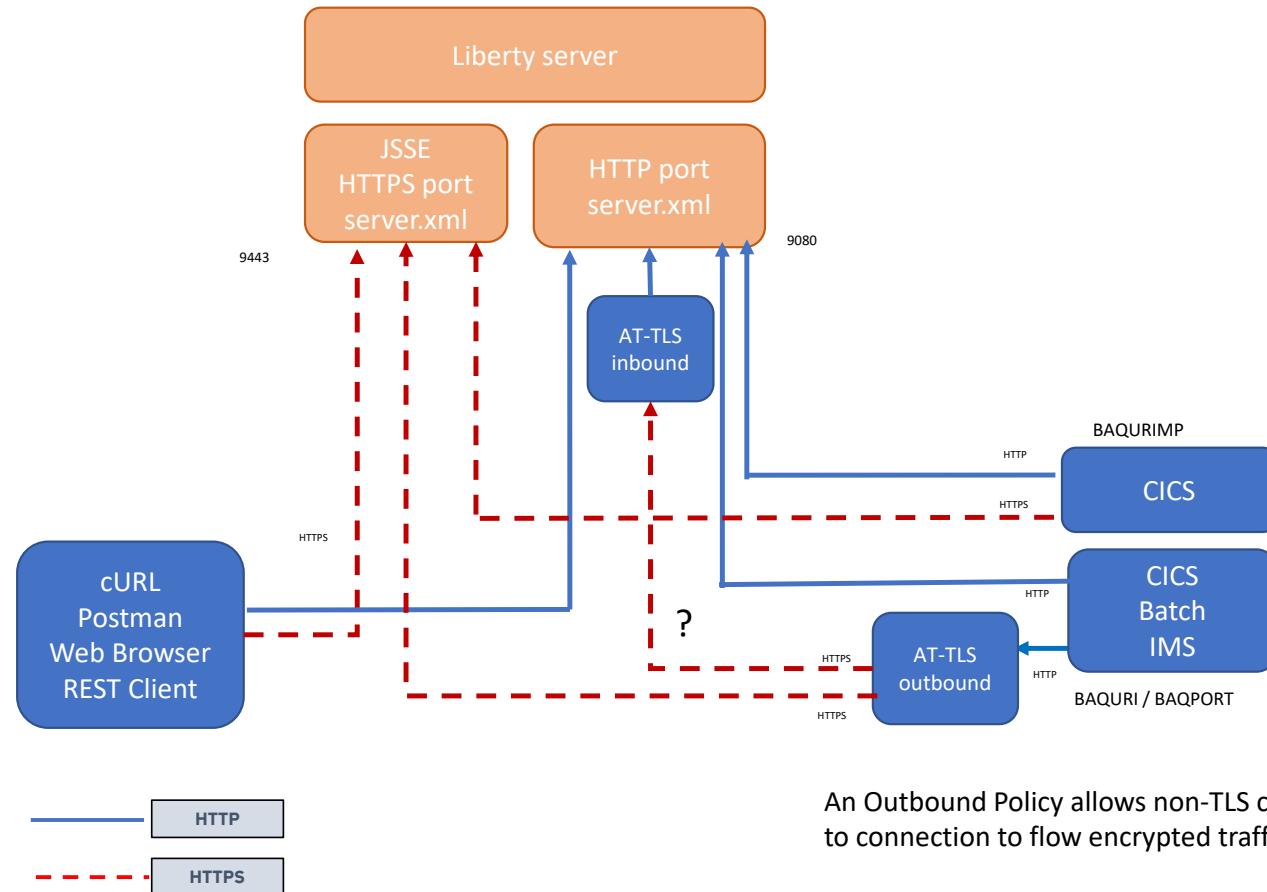
The server XML configuration uses no HTTPS protocol, key rings or other JSSE attributes



- **Application Transparent TLS (AT-TLS)** creates a secure session on behalf of z/OS Connect
- Only define http ports in **server.xml** (z/OS Connect does not know that TLS session exists)
- Define TLS protection for all applications (including z/OS Connect) in **AT-TLS policy**
- AT-TLS uses **System SSL** which exploits the CPACF and Crypto Express cards via ICSF

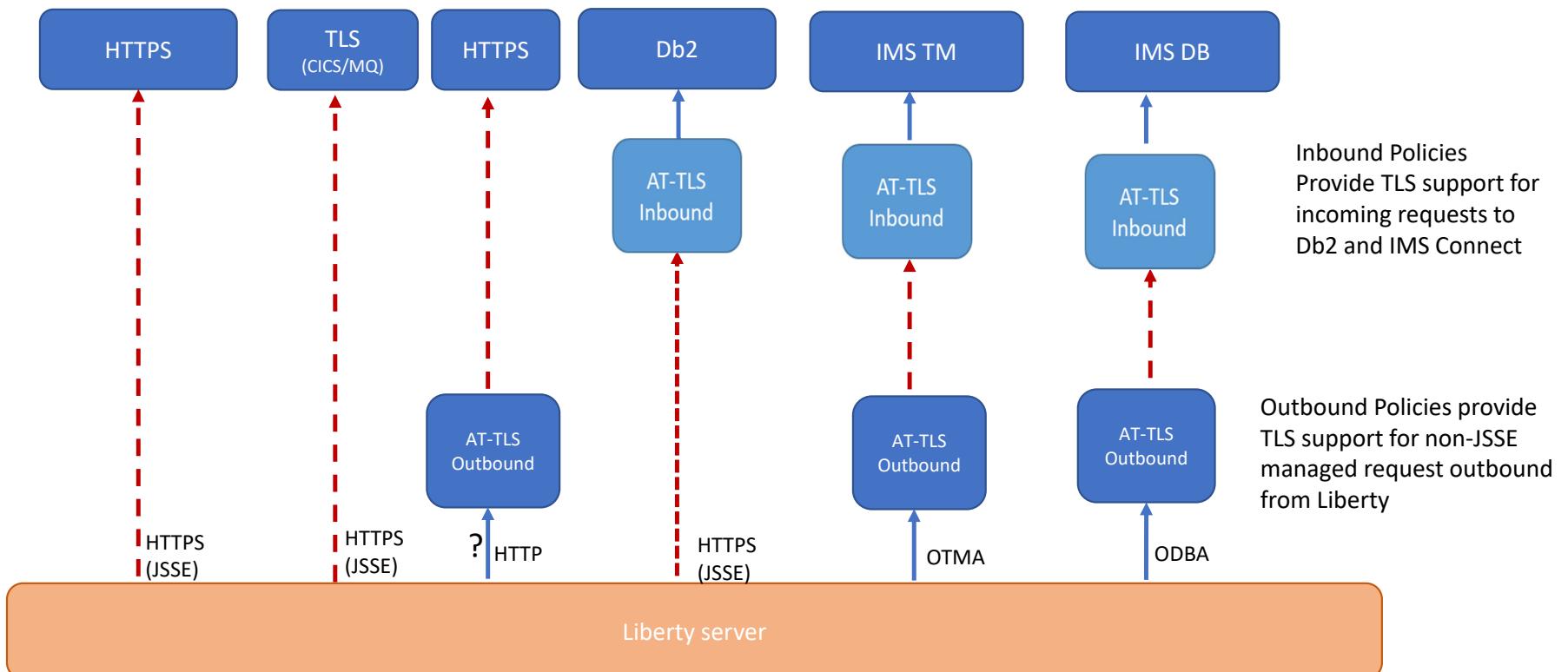


TLS client encryption to a Liberty server scenarios





TLS encryptions from a Liberty server (HTTPS/native to HTTPS/TLS/OTMA/ODBA)



**Let's explore using TLS for
encryption and data integrity
using samples in various scenarios**



Using this Liberty JSSE server XML configuration

```
<!-- Enable features -->
<featureManager>
    <feature>transportSecurity-1.0</feature>
</featureManager>

<sslDefault sslRef="DefaultSSLSettings"
    outboundSSLRef="OutboundSSLSettings" />

<ssl id="DefaultSSLSettings"
    keyStoreRef="CellDefaultKeyStore"
    trustStoreRef="CellDefaultKeyStore"
    clientAuthenticationSupported="true"
    clientAuthentication="true"
    serverKeyAlias="Liberty Server Cert"/>

<keyStore id="CellDefaultKeyStore"
    location="safkeyring:///Liberty.KeyRing"
    password="password" type="JCERACFKS"
    fileBased="false" readOnly="true" />

<ssl id="OutboundSSLSettings"
    keyStoreRef="OutboundKeyStore"
    trustStoreRef="OutboundKeyStore"
    clientKeyAlias="Liberty Client Cert"/>

<keyStore id="OutboundKeyStore"
    location="safkeyring:///zCEE.KeyRing"
    password="password" type="JCERACFKS"
    fileBased="false" readOnly="true" />

<zosconnect_authorizationServer sslCertsRef="SSL repertoire"/>
<zosconnect_cicsIpicConnection sslCertsRef="SSL repertoire"/>
<zosconnect_endpointConnect sslCertsRef="SSL repertoire"/>
<zosconnect_zosConnectRestClient sslCertsRef="SSL repertoire"/>
<zosconnect_zosConnectServiceRestClientConnection sslCertsRef="SSL repertoire"/>
```

SSL repertoires

F BAQSTRT,REFRESH,KEYSTORE

Let's explore TLS options using the contents of these key rings

Liberty's outbound key ring

Digital ring information for user LIBSERV:			
Ring: >zCEE.KeyRing<			
Certificate Label Name	Cert Owner	USAGE	DEFAULT
zCEE CA	CERTAUTH	CERTAUTH	NO
Liberty CA	CERTAUTH	CERTAUTH	NO
zCEE Client Cert	ID (LIBSERV)	PERSONAL	YES
xyz Client Cert	ID (LIBSERV)	PERSONAL	NO
DB2 CA	CERTAUTH	CERTAUTH	NO
MQ CA	CERTAUTH	CERTAUTH	NO
CICS CA	CERTAUTH	CERTAUTH	NO

Ring: >Liberty.KeyRing<			
Certificate Label Name	Cert Owner	USAGE	DEFAULT
Liberty Server Cert	ID (LIBSERV)	PERSONAL	YES
Liberty CA	CERTAUTH	CERTAUTH	NO
zCEE CA	CERTAUTH	CERTAUTH	NO
CICS CA	CERTAUTH	CERTAUTH	NO

Digital ring information for user CICSSTC:			
Ring: >CICS.KeyRing<			
Certificate Label Name	Cert Owner	USAGE	DEFAULT
CICS CA	CERTAUTH	CERTAUTH	NO
CICS Client Cert	ID (CICSSTC)	PERSONAL	YES
Liberty CA	CERTAUTH	CERTAUTH	NO
zCEE CA	CERTAUTH	CERTAUTH	NO

Digital ring information for user DB2USER:			
Ring: >Db2.KeyRing<			
Certificate Label Name	Cert Owner	USAGE	DEFAULT
DB2 CA	CERTAUTH	CERTAUTH	NO
zCEE CA	CERTAUTH	CERTAUTH	NO
DB2USER	ID (DB2USER)	PERSONAL	YES

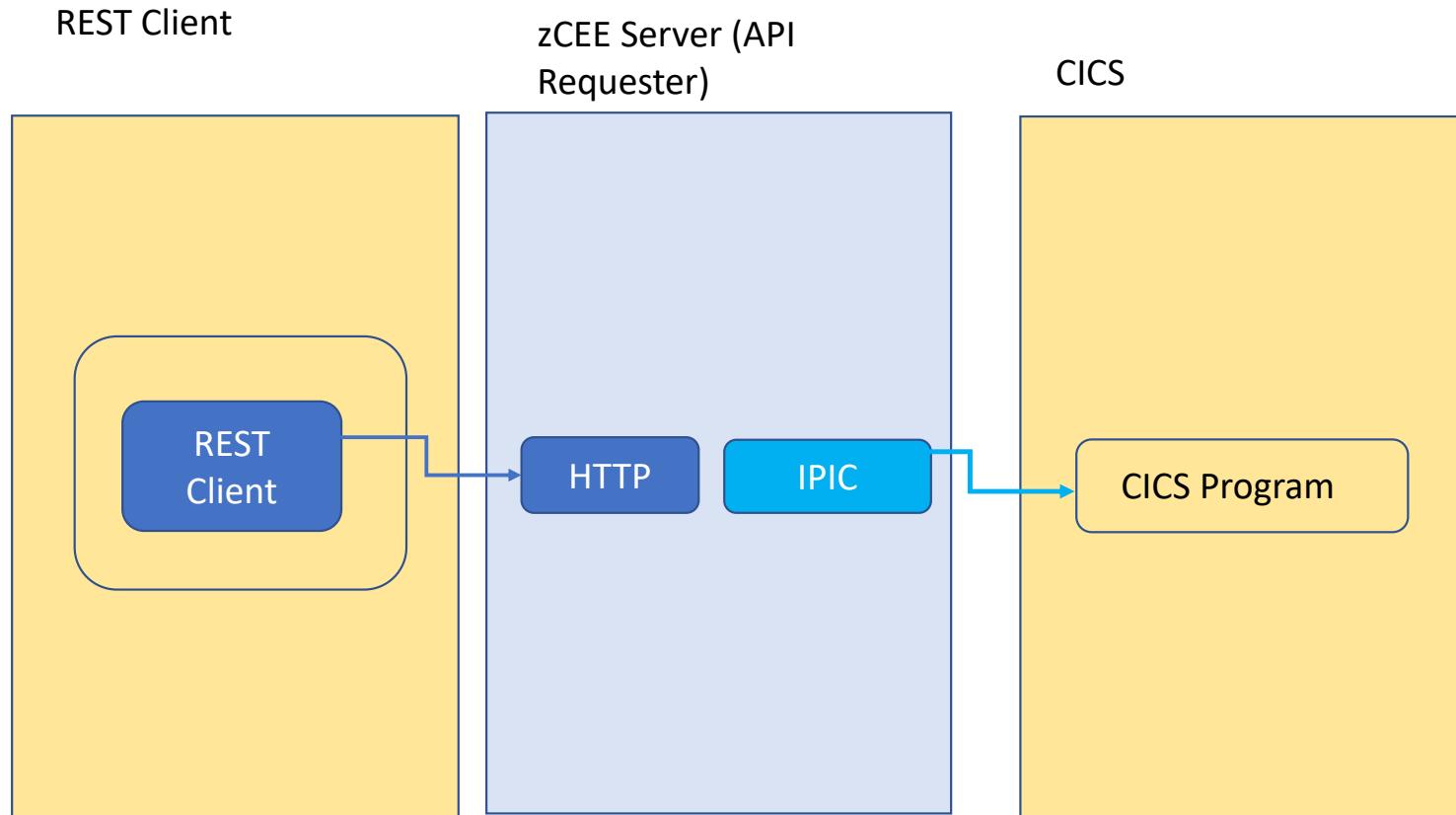
Tech-Tip: when Liberty is the client endpoint, and more than one personal certificate is connected to a key ring. Use the SSL repertoire *clientKeyAlias* attributes to select the personal certificate to be used in a handshake.

Liberty's inbound key ring

Tech-Tip: when Liberty is the server endpoint, and more than one personal certificate is connected to a key ring. Use the SSL repertoire *serverKeyAlias* attributes to select the personal certificate to be used in a handshake.

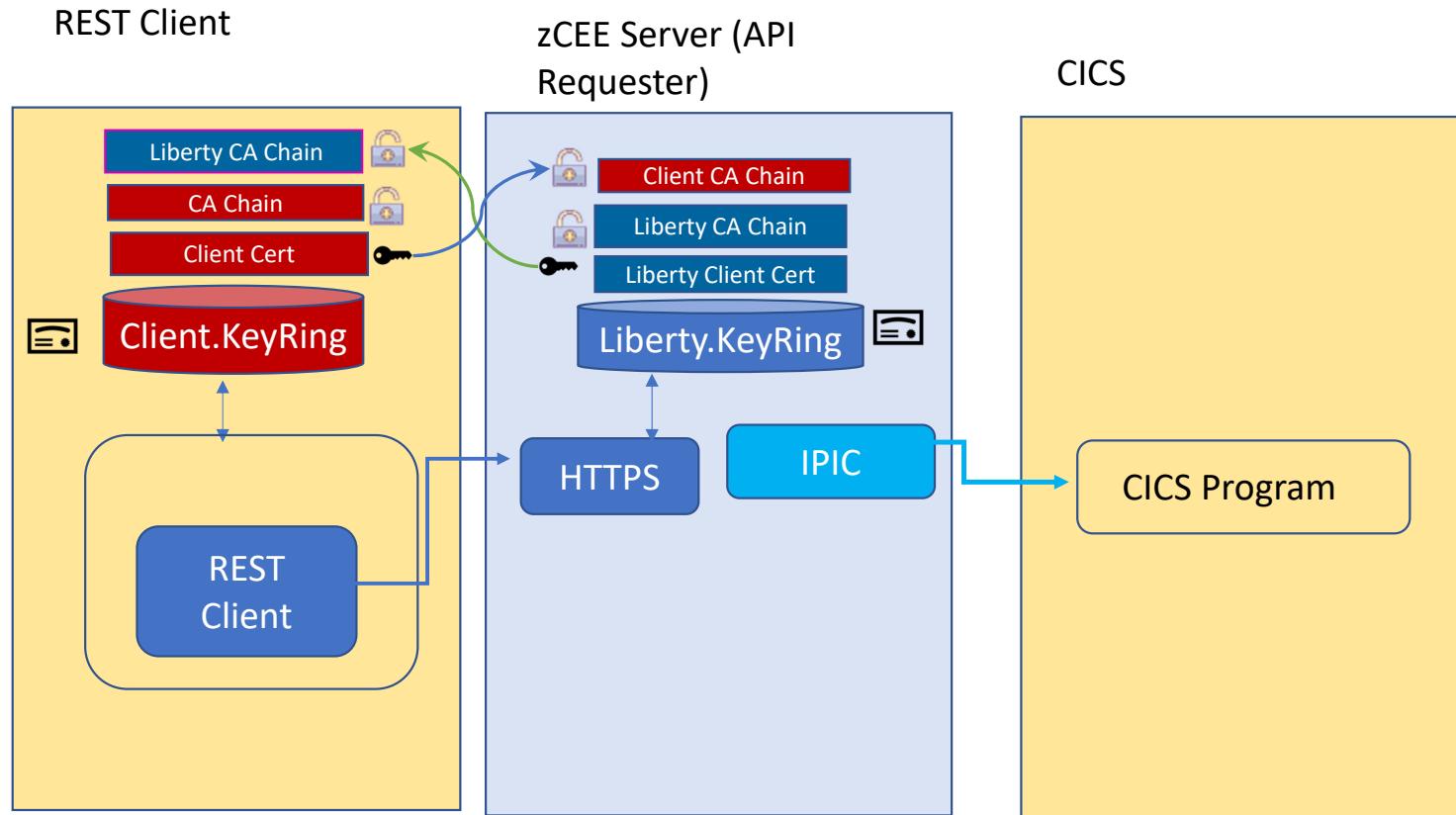


No TLS between any endpoints



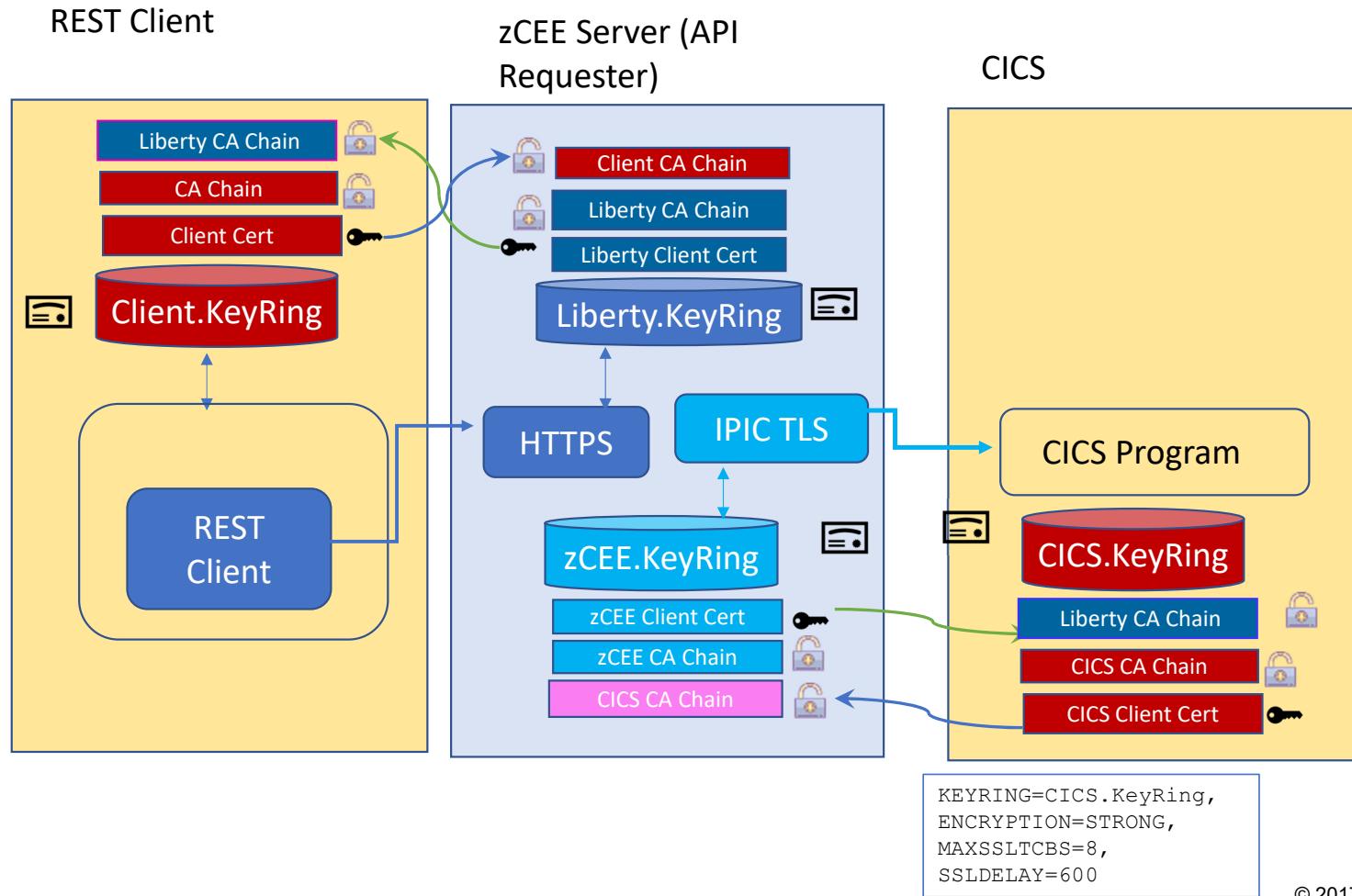


TLS handshake between the client and z/OS Connect server



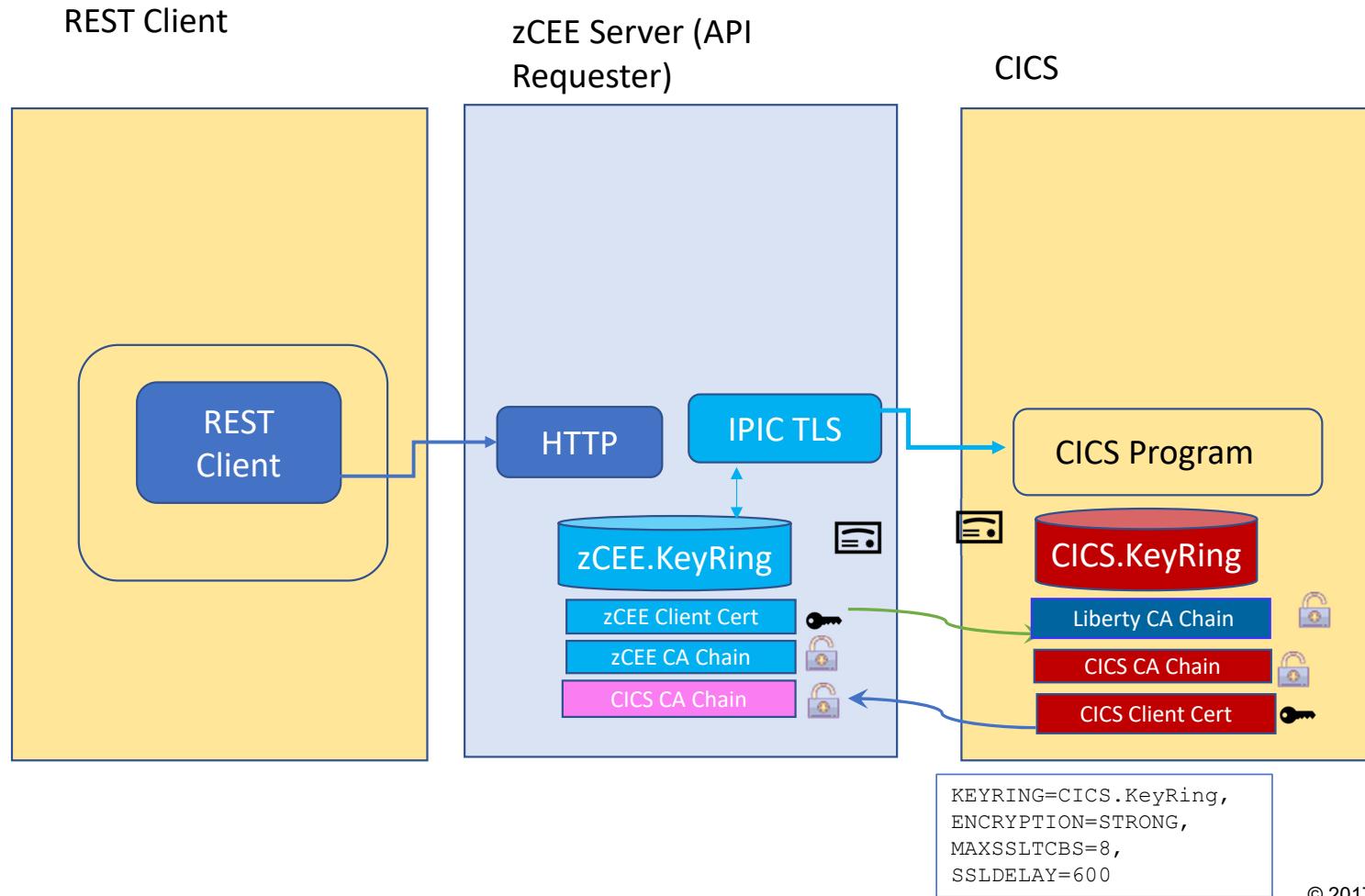


Independent TLS handshakes between all endpoints





TLS handshake between the z/OS Connect server and the target endpoint





CICS IPIC using TLS

The server.xml file is the key configuration file:

inquireSingle Service

Configuration

Required Configuration

Enter the required configuration for this service.

Coded character set identifier (CCSID):

Connection reference:

Optional Configuration

Enter the optional configuration for this service.

Transaction ID:

Transaction ID usage:

Overview Configuration

Liberty Admin Center

Server Config

ipicSSLIDProp.xml

Read only Close

```

<server description="CICS IPIC ID propagation connections">
  <!-- Enable features -->
  <featureManager>
    <feature>zosconnect:cicsService-1.0</feature>
  </featureManager>
  <zosconnect_cicsIpicConnection id="catalog"
    host="wg31.washington.ibm.com"
    port="1493"
    zosConnectNetworkid="CSCVINC"
    zosConnectApplid="CSCVINC"
    transid="M10"
    transidUsage="EIB_AND_MIRROR"
    sslCertsRef="cicsSSLSettings"/>
</server>

```

WG31

File Edit Settings View Communication Actions Window Help

I TCPIPS
RESULT - OVERTYPE TO MODIFY
Tcpipservice(CSCVINC)
Openstatus(Open)
Port(01493)
Protocol(Ipi...
Seltype(Ssl)
Transid(CISS)
Authenticate(NoAuthentic)
Connections(00000)
Backlog(01024)
Maxdatalen(000000)
Urm(DFHISAPI)
Privacy(Supported)
Ciphers(35388392F3233)
Host(ANY)
IpAddress(192.168.17.201)
Hosttype(Any)
Ipresolved(192.168.17.201)
+ Ipfamily(Ipv4family)

SYSID=CICS APPLID=CICS53Z
TIME: 13.12.07 DATE: 02/22/21
PF 1 HELP 2 HEX 3 END 5 VAR 7 SBH 8 SFH 10 SB 11 SF 01/012
MB D
Connected to remote server/host wg31 using lu/pool TCP00137 and port 23

Define IPIC/TLS connections to CICS

Tech/Tip: Cipher Suite numbers (CICS TCPIPSERVICE):

Table 2. 2-character and 4-character cipher suite definitions for SSL V3, TLS V1.0, TLS V1.1, TLS V1.2, and TLS V1.3

2-character cipher number	4-character cipher number	Short name	Description ¹	FIPS 140-2	Base security level	Security level 3 FMID > JCPT441 <
00	0000	TLS_NULL_WITH_NULL_NULL	No encryption or message authentication and RSA key exchange	X	X	
01	0001	TLS_RSA_WITH_NULL_MD5	No encryption with MD5 message authentication and RSA key exchange	X	X	
02	0002	TLS_RSA_WITH_NULL_SHA	No encryption with SHA-1 message authentication and RSA key exchange	X	X	
03	0003	TLS_RSA_EXPORT_WITH_RC4_40_MD5	40-bit RC4 encryption with MD5 message authentication and RSA (export) key exchange	X	X	
04	0004	TLS_RSA_WITH_RC4_128_MD5	128-bit RC4 encryption with MD5 message authentication and RSA key exchange		X	
05	0005	TLS_RSA_WITH_RC4_128_SHA	128-bit RC4 encryption with SHA-1 message authentication and RSA key exchange	X		^

https://www.ibm.com/support/knowledgecenter/SSLTBW_2.4.0/com.ibm.zos.v2r4.gska100/csdcwh.htm



MQ JMS using TLS

The server.xml file is the key configuration file:

The diagram illustrates the configuration of an MQ JMS service using TLS across three different interfaces:

- Service Project Editor: Configuration**: Shows the "Required Configuration" section with fields like Connection factory JNDI name (jms/qmgrCf), Request destination JNDI name (jms/requestQueue), Reply destination JNDI name (jms/replyQueue), Wait interval (3000), MQMD format (MQSTR), Coded character set identifier (CCSID) (37), Is message persistent (unchecked), Reply selection (msgIDToCorrelID), and Expiry (-1).
- Server Config**: Displays the XML configuration file `mqClientTLS.xml`. The `<jmsConnectionFactory id="qmgrCf" jndiName="jms/qmgrCf">` section is highlighted with a red oval, specifically the `sslCipherSuite="SSL_RSA_WITH_AES_256_CBC_SHA256"` attribute.
- LIBERTY.SSL.SVRCONN - Properties**: Shows the SSL tab with the Cipher Spec set to `TLS_RSA_WITH_AES_256_CBC_SHA256`.

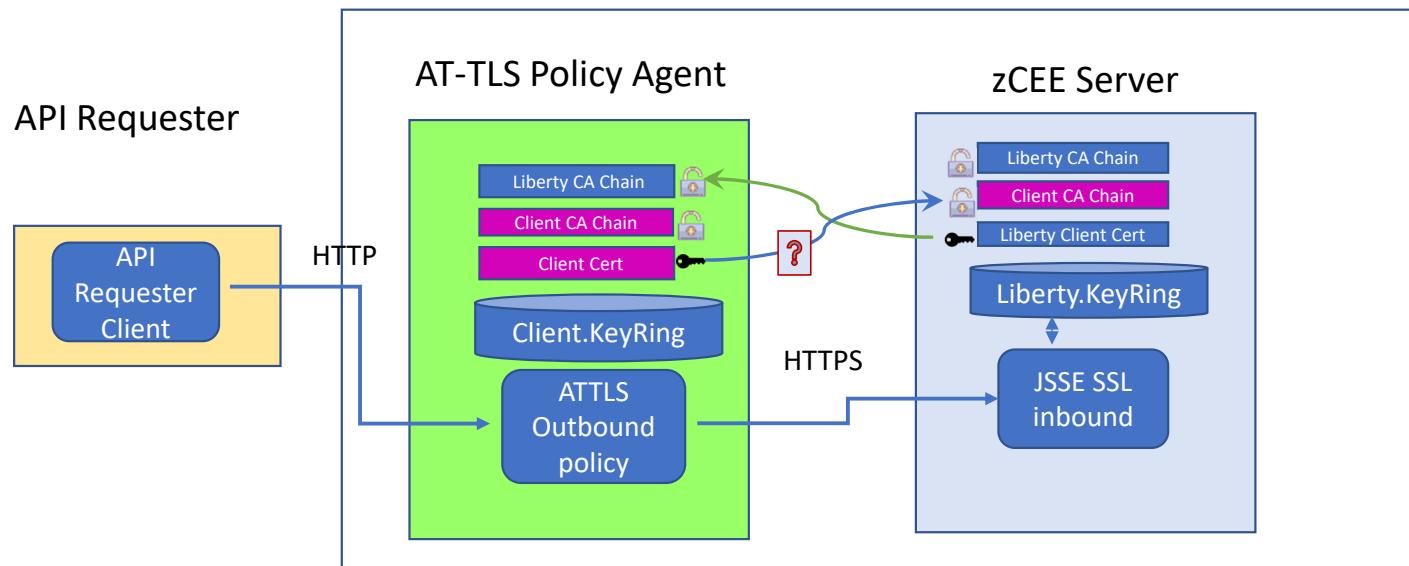
Dotted arrows connect the configuration values in each interface to the corresponding XML elements in the `mqClientTLS.xml` file, demonstrating how the visual settings map to the underlying XML structure.

```
1 <server description="MQ Service Provider">
2   <featureManager>
3     <feature>zosconnect:mqService-1.0</feature>
4   </featureManager>
5 
6   <variable name="wmqJmsClient.rar.location"
7     value="/u/johnson/jca/wmq.jmsra.rar"/>
8   <wmqJmsClient nativeLibraryPath="/usr/lpp/mqm/V9R1M1/java/lib"/>
9 
10  <zosconnect_services>
11    <service name="mqPutService">
12      <property name="useCallerPrincipal" value="true"/>
13    </service>
14  </zosconnect_services>
15 
16  <connectionManager id="ConMgr1" maxPoolSize="5"/>
17 
18  <jmsConnectionFactory id="qmgrCf" jndiName="jms/qmgrCf">
19    connectionManagerRef="ConMgr1">
20      <properties.wmqJMS transportType="CLIENT"
21        queueManager="ZMQ1"
22        channel="LIBERTY.SSL.SVRCONN"
23        hostName="wg31.washington.ibm.com"
24        sslCipherSuite="SSL_RSA_WITH_AES_256_CBC_SHA256"
25        port="1422"/>
26    </jmsConnectionFactory>
27 
28  <jmsQueue id="q1" jndiName="jms/default">
29    <properties.wmqJMS
30      baseQueueName="ZCEE.DEFAULT.MQZCEE.QUEUE"
31      CCSID="37"/>
32  </jmsQueue>
33 
34 </server>
```



AT-TLS - outbound policy handshake scenarios

Policy Agent uses an outbound policy and acts a surrogate TLS client



🔑 Certificate with a private key

🔒 Certificate Authority (CA) certificate

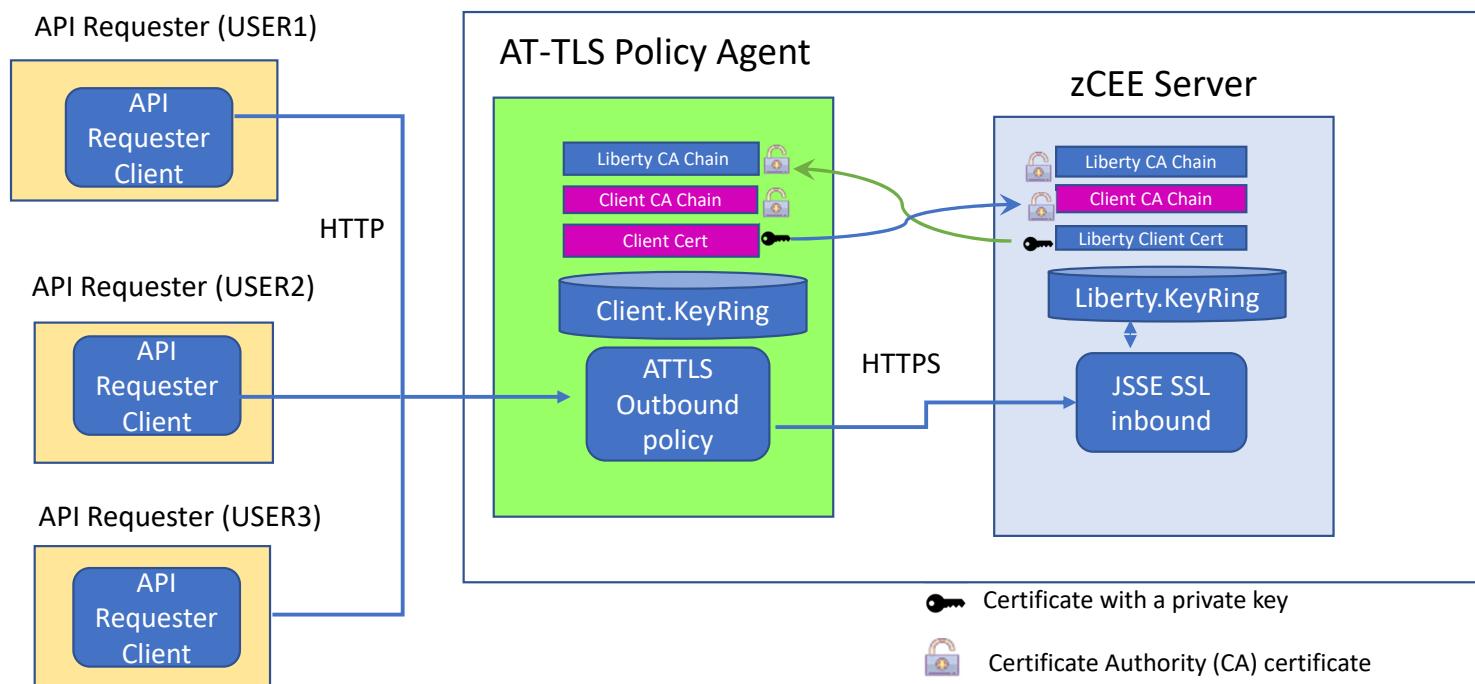
```
<zosconnect_apiRequesters idAssertion="ASSERT_ONLY">  
</zosconnect_apiRequesters>
```



Question if this really needed, remember TLS encryption is independent of TLS authentication.

AT-TLS - outbound policy handshake scenario

Use of a common key ring name for multiple client identities

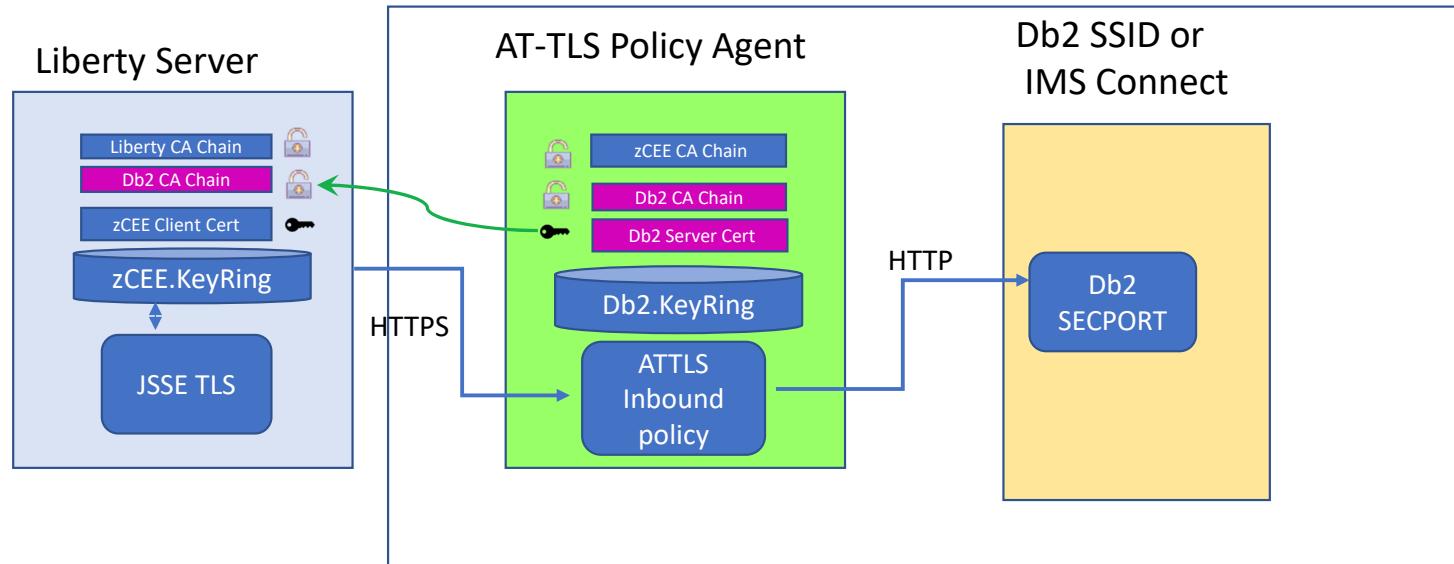


- Each user owns a keyring with the name Liberty.KeyRing.
- Each key ring has a different default client certificate for mutual authentication purposes.

This is a situation when AT-TLS mutual authentication has a benefit.

AT-TLS - inbound policy handshake scenario (Db2 and IMS)

Policy Agent uses both inbound and outbound policies and acts a surrogate TLS client with one and a TLS server with the other



Note that DB2 is AT-TLS aware
IMS is AT-TLS unaware

🔑 Certificate with a private key

🔒 Certificate Authority (CA) certificate



API Requester - HTTP v HTTPS

MVS Batch and IMS with and without an outbound AT-TLS policy

```
CEE0PTS DD *
  POSIX(ON),
  ENVAR("BAQURI=wg31.washington.ibm.com",
  "BAQPORT=9080")
```

```
CEE0PTS DD *
  POSIX(ON),
  ENVAR("BAQURI=wg31.washington.ibm.com",
  "BAQPORT=9443")
```

CICS URIMAPS

The image shows two side-by-side CICS URIMAPS configuration panels, both titled 'WG31'. The left panel is for 'CICS RELEASE = 0710' and the right panel is for 'CICS RELEASE = 0710'. Both panels display the 'OVERTYPE TO MODIFY' command for altering a Urimap named 'BAQURIMP'. The configuration parameters shown are:

Parameter	Left Host (Value)	Right Host (Value)
Urimap	BAQURIMP	BAQURIMP
Group	SYSGRP	SYSGRP
DESCription	URIMAP for z/OS Connect EE server	URIMAP for z/OS Connect EE server
Status	Enabled	Enabled
USAge	Client	Client
	Server Client Pipeline Atom	Server Client Pipeline Atom
	Jvmserver	Jvmserver
UNIVERSAL RESOURCE IDENTIFIER		
SCHEME	HTTP HTTPS	HTTP HTTPS
POrt	09120	09443
HOST	wg31.washington.ibm.com	wg31.washington.ibm.com
Path	/	/
(Mixed Case)	==>	==>
	==>	==>
	==>	==>
	==>	==>
+ OUTBOUND CONNECTION POOLING		
	SYSID=CICS APPL	SYSID=CICS APPLID=CICSS3Z
PF 1 HELP 2 COM 3 END	6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11	6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
MBP C		13/022
Connected to remote server/host wg31 using lu/pool TCP00133 and port 23		

Field BAQ-ZCON-SERVER-URI was added to BAQRINFO in V3.0.37.

MOVE "URIMAP01" TO BAQ-ZCON-SERVER-URI.

mitchj@us.ibm.com



Connection Management

- Amount of time before a connection can be discarded by pool maintenance.
- Amount of time after which a connection request times out.
- Amount of time a connection can be unused or idle until it can be discarded during pool maintenance.
- Maximum number of physical connections for a pool.
- Minimum number of physical connections to maintain in the pool.
- Specifies which connections to destroy when a stale connection is detected in a pool.
- Amount of time between runs of the pool maintenance thread.

```
<connectionManager id="ConMgr1"  
agedTimeout=-1  
connectionTimeout=30  
maxIdleTime=1800  
maxPoolSize=50  
minPoolSize=0  
purgePolicy="EntirePool"  
reapTime=180/>
```

Ciphers



- During the TLS handshake, the TLS protocol and data exchange cipher are negotiated
- Choice of cipher and key length has an impact on performance
- You can restrict the protocol (TLS) and ciphers to be used
- Example setting server.xml file

```
<ssl id="DefaultSSLSettings" keyStoreRef="defaultKeyStore"  
sslProtocol="TLSv1.2"  
enabledCiphers="TLS_RSA_WITH_AES_256_CBC_SHA256  
TLS_RSA_WITH_AES_256_GCM_SHA384"/>
```

- This configures use of TLS 1.2 and two supported ciphers
- It is recommended to control what ciphers can be used in the server rather than the client

For cipher details, see IBM SDK Java 8.0.0 Cipher Suites at URL

https://www.ibm.com/support/knowledgecenter/SSYKE2_8.0.0/com.ibm.java.security.component.80.doc/security-component/jsse2Docs/ciphersuites.html



Persistent connections

- Persistent connections can be used to avoid too many handshakes
- Configured by setting the `keepAliveEnabled` attribute on the `httpOptions` element to **true**
- Example setting `server.xml` file

```
<httpEndpoint host="*" httpPort="80" httpsPort="443" id="defaultHttpEndpoint"
httpOptionsRef="httpOpts"/>

<httpOptions id="httpOpts" keepAliveEnabled="true" maxKeepAliveRequests="500"
persistTimeout="1m"/>
```

- This sets the connection timeout to **1 minute** (default is 30 seconds) and sets the maximum number of persistent requests that are allowed on a single HTTP connection to **500**
- It is recommended to set a maximum number of persistent requests when connection workload balancing is configured
- It is also necessary to configure the client to support persistent connections



TLS sessions

- When connections timeout, it is still possible to avoid the impact of full handshakes by reusing the TLS session id
- Configured by setting the `sslSessionTimeout` attribute on the `sslOptions` element to an amount of time
- Example setting `server.xml` file

```
<httpEndpoint host="*" httpPort="80" httpsPort="443" id="defaultHttpEndpoint"  
httpOptionsRef="httpOpts" sslOptionsRef="mySSLOptions"/>  
  
<httpOptions id="httpOpts" keepAliveEnabled="true" maxKeepAliveRequests="100"  
persistTimeout="1m"/>  
  
<sslOptions id="mySSLOptions" sslRef="DefaultSSLSettings"  
sslSessionTimeout="10m"/>
```

- This sets the timeout limit of an TLS session to **10 minutes** (default is 8640ms)
- TLS session ids are not shared across z/OS Connect servers

Enabling hardware cryptography key rings

jvm.options

```
-Djava.security.properties=${server.config.dir}/java.security
```

java.security

```
security.provider.1=com.ibm.crypto.hdwrCCA.provider.IBMJCECCA  
security.provider.2=com.ibm.crypto.provider.IBMJCE  
security.provider.3=com.ibm.jsse2.IBMJSSEProvider2  
security.provider.4=com.ibm.security.jgss.IBMJGSSProvider  
.....
```

Enabling the IBMJCECCA provider

```
<keyStore id="CellDefaultKeyStore"  
location="safkeyringhw://Liberty.KeyRing"  
password="password" type="JCECCARACFKS"  
fileBased="false" readOnly="true" />
```

Enabling the IBMJCEHYBRID provider

```
<keyStore id="CellDefaultKeyStore"  
location="safkeyringhybrid://Liberty.KeyRing"  
password="password" type="JCEHYBRIDRACFKS"  
fileBased="false" readOnly="true" />
```

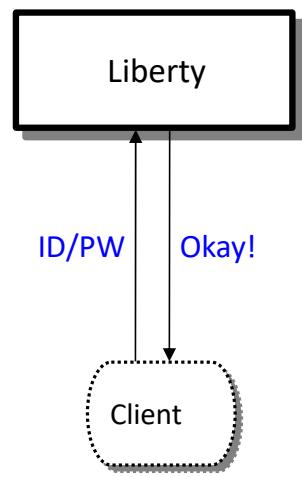
See URL <https://www.ibm.com/support/pages/node/6209109> for details on implementing IBMJCECCA and IBMJCEHYBRID hardware encryption providers



Authentication - Third Party Authentication

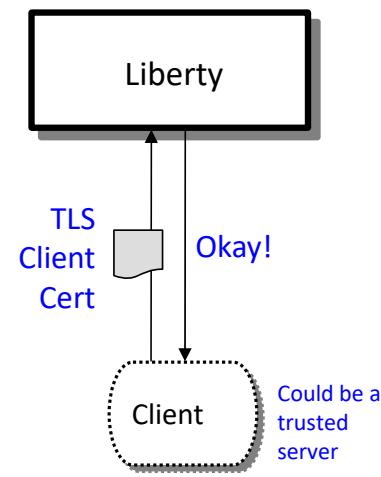
Several different ways this can be accomplished:

Basic Authentication



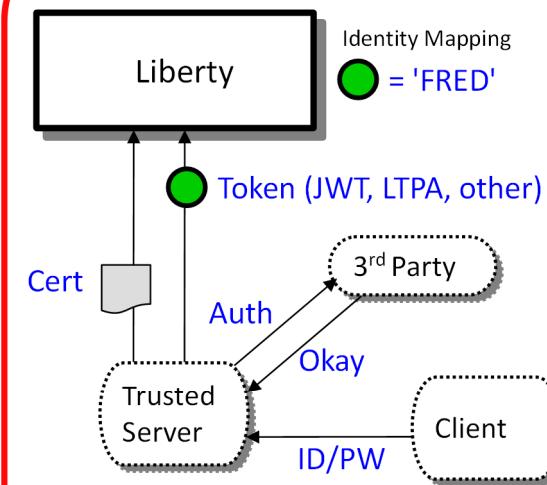
- Server prompts for ID/PW
- Client supplies ID/PW or ID/PassTicket
- Server checks registry:
 - Basic (server.xml)
 - SAF

Client Certificate



- Server prompts for client certificate.
- Client supplies certificate
- Server validates client certificate and maps to an identity
- Registry options:
 - SAF

Third Party Authentication



- Client authenticates to 3rd party sever
- Client receives a trusted 3rd party token
- Token flows to Liberty z/OS and is mapped to an identity
- Registry options:
 - We may know these detail.



Third Party Authentication Examples

The image displays two side-by-side screenshots of web pages illustrating third-party authentication.

Left Screenshot: UPS Sign Up

This screenshot shows the UPS Sign Up page. At the top, there's a banner stating "UPS is open for business: Service impacts related to Coronavirus ...More". Below the banner, the UPS logo is displayed. A "Sign Up / Log in" link and a "Search or Track" input field are visible. The main section is titled "Sign Up" and includes a link for users who already have an ID. It provides several options for sign-up via external services: Google, Facebook, Amazon, Apple, and Twitter. Below these, fields for "Name *", "Email *", "User ID *", "Password *", and "Phone" are provided. The "Password *" field has a "Show" link next to it. A "Feedback" button is located on the right side of the form.

Right Screenshot: myNCDMV Log In

This screenshot shows the myNCDMV Log In page. The background features a scenic view of autumn foliage. At the top, there are "Log In" and "Sign Up" links. The main area is titled "Log In to myNCDMV" and contains fields for "Email Address" (with "name@example.com" entered) and "Password" (with "*****" entered). A "Remember Me" checkbox is available. Below the password field are "Log In" and "Forgot Password" buttons. Further down, there are three social media sign-in options: "Continue with Apple", "Continue with Facebook", and "Continue with Google". A "Continue as Guest" link is also present. A notice at the bottom states: "NOTICE FOR PUBLIC COMPUTER USERS - If you sign in with Google, Apple, or Facebook you are also signing into that account on this computer. Remember to sign out when you're done." The page is powered by "payit".



Open security standards

- **OAuth** is an open standard for access delegation, used as a way to grant websites or applications access to their information without requiring a password.
- **OpenID Connect** is an authentication layer on top of OAuth. It allows the verification of the identity of an end-user based on authentication performed by an authorization server.
- **JWT (JSON Web token)** defines a compact and self-contained way for securely transmitting information between parties as a JSON object

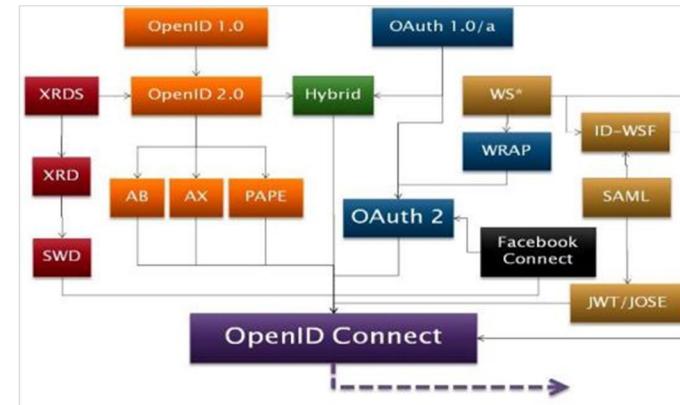
See the YouTube videos:

OAuth 2.0 and OpenID Connect (in plain English)

<https://www.youtube.com/watch?v=996OjexHze0>

OpenID Connect on Liberty

<https://www.youtube.com/watch?v=fuajCS5bG4c>



© 2017, 2022 IBM Corporation

What is a JWT (JSON Web Token) ?

- JWT is a compact way of representing claims that are to be transferred between two parties
- Normally transmitted via HTTP header
- Consists of three parts
 - Header
 - Payload
 - Signature

The screenshot shows the jwt.io debugger interface. At the top, it says "Encoded" and displays a long string of characters: eyJraWQi0iI0cWpYLWJrWE9Vd19GX... The bottom right corner of this string has a red oval highlighting the timestamp "Mon Nov 02 2020 11:05:58 GMT-0500 (Eastern Standard Time)". To the right, under "Decoded", there are two sections: "HEADER:" and "PAYLOAD:". The "HEADER:" section contains a JSON object with "kid": "4qjX-bkX0Uw_F_uccjRMkB9ivMjXSQwj0RrkYRJq8DM" and "alg": "RS256". The "PAYLOAD:" section contains a JSON object with "sub": "Fred", "token_type": "Bearer", "scope": ["openid", "profile", "email"], "azp": "rpSsl", "iss": "https://wg31.washington.ibm.com:26213/oidc/endpoint/0P", "aud": "myZcee", "exp": 1604333158, "iat": 16043330858, "realmName": "zCEERealm", and "uniqueSecurityName": "Fred".

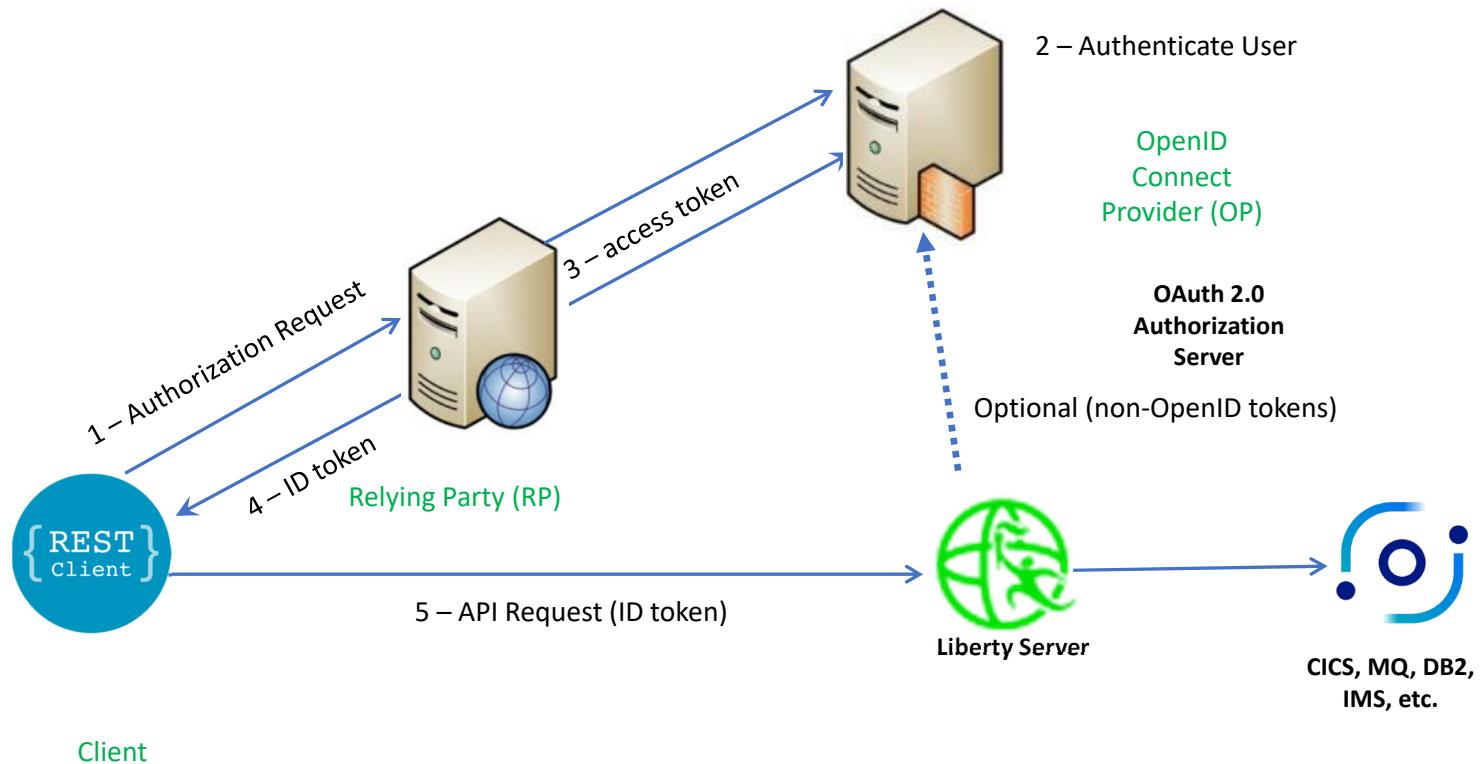
Values derived from the OAUTH configuration:

- signatureAlgorithm="RS256"
- accessTokenLifetime="300"
- resourceIds="myZcee"

<https://jwt.io>



Typical Authorization Flow for an OpenID Connect token to a z/OS Connect API Provider



OpenID Connect/OAuth and z/OS Connect

- **From the z/OS Connect Knowledge Center:** z/OS Connect EE security can operate with traditional z/OS security, for example, System Authorization Facility (SAF) and also with open standards such as Transport Layer Security (TLS), JSON Web Token (JWT), and **OpenID Connect**.
- **From the OpenID Core specification:** OpenID Connect 1.0 is a simple identity layer on top of the OAuth 2.0 protocol. It enables Clients to verify the identity of the End-User based on the authentication performed by an Authorization Server, as well as to obtain basic profile information about the End-User in an interoperable and REST-like manner.
- **OAuth 2.0 Core (RFC 6749) Specifications:** <https://tools.ietf.org/html/rfc6749>
- **OpenID Connect Core Specifications:** https://openid.net/specs/openid-connect-core-1_0.html
- **Again, for a very good explanation of this topic see YouTube video OAuth 2.0 and OpenID Connect (in plain English)**
<https://www.youtube.com/watch?v=996OjexHze0>

Some basic OAuth/OpenID Connect terms

- **Authorization server** - The server that issues access tokens to the client after authenticating the resource owner and obtaining authorization. *In a z/OS Connect EE API requester scenario, the authorization server is called by the z/OS Connect EE server to retrieve an access token.*
- **Authorization Endpoint** - A service or endpoint on an OAuth authorization server that accepts an authorization request from a client to perform authentication and authorization of a user. The authorization endpoint returns an authorization grant, or code, to the client in the Authorization Code Flow. In the Implicit Flow, the authorization endpoint returns an access token to the client.
- **Token Endpoint** – A service or endpoint on an OP that accepts an authorization grant, or code, from a client in exchange for an access token, ID token, and refresh token
- **Access Token** – A credential that is used to access protected resources. An access token is a string that represents an authorization that is issued to the client. The access token is usually opaque to the client (it does not have to be opaque) and can be JSON Web Token (JWT). See URL <https://tools.ietf.org/html/rfc6749> Section 1.4 for more information.
- **OAuth token** - With OAuth 2.0, access tokens are used to access protected resources. An access token is normally a string that represents an authorization that is issued to the client. The string is usually opaque to the client. Opaque tokens may require that the token recipient call back to the server that issued the token. *However, an access token can also be in the form of a JSON Web Token (JWT) which does not require a call back (introspection).*
- **Scope** - Privilege or permission that allows access to a set of resources of a third party.

Some basic OAuth/OpenID Connect terms

- **Relying Party (RP)** – An entity that relies on an OP to authenticate a user and obtain an authorization to access a user's resource.
For z/OS Connect API Requester, it is the Liberty server configured as an OpenID Connect Client, e.g., using <openidConnectClient> XML configuration elements.
- **OpenID Connect Provider (OP)** - An OAuth 2.0 authorization server that is capable of providing claims to a client or Relying Party (RP) , *an OpenID component.*
- **Resource owner** - An entity capable of granting access to a protected resource. When the resource owner is a person, it is referred to as an end user. *In a z/OS Connect EE API requester scenario, the resource owner might be the user of the CICS, IMS, or z/OS application.*
- **Resource server** - The server that hosts the protected resources and accepts and responds to protected resource requests by using access tokens. *In a z/OS Connect API provider, the resource server is the z/OS Connect server. In a z/OS Connect EE API requester scenario, the resource server is the request endpoint for the remote RESTful API*
- **ID Token** - is an OpenID Connect token that is an extension to OAuth 2.0 specification access tokens. This token is a JSON Web Token (JWT). See URL https://openid.net/specs/openid-connect-core-1_0.html#IDToken for more information about the extensions.



Tech/Tip: Let's explore a flow using a Liberty OpenID Provider as an example

This Liberty server configuration provides a good example of the workings of an authorization server.

```
<httpEndpoint host="*" httpPort="26212" httpsPort="26213" id="defaultHttpEndpoint"/>

<openidConnectProvider id="OP"
    signatureAlgorithm="RS256"
    keyStoreRef="jwtStore"
    oauthProviderRef="OIDCssl" >
</openidConnectProvider>

<oauthProvider id="OIDCssl"
    httpsRequired="true"
    jwtAccessToken="true"
    autoAuthorize ="true"
    accessTokenLifetime="300">

    <!-- Define OIDC Client for zCEE Authentication -->
    <autoAuthorizeClient>zCEEClient</autoAuthorizeClient>
    <localStore>
        <client name="zCEEClient"
            secret="secret"
            displayname="zCEEClient"
            scope="openid"
            enabled="true"
            resourceIds="myZcee"/>
    </localStore>
</oauthProvider>
```

Key Points:

- **keyStoreRef** - A keystore containing the private key necessary for signing with an asymmetric algorithm.
- **jwtAccessToken** - generate a JSON Web Token, serialize it as a string and put in the place of the access token.

Tech/Tip: Generating a JWT using Liberty's as an example OPID provider



The Liberty server authorization server's XML configuration

```
<!--Key store that contains certificate used to sign JWT-->
<keyStore fileBased="false" id="jwtStore"
  location="safkeyring:///JWT.KeyRing"
  password="password" readOnly="true" type="JCERACFKS"/>

<!-- Define a basic user registry -->
<basicRegistry id="basicRegistry"
  realm="zCEERealm">
  <user name="auser" password="pwd"/>
  <user name="distributed_User1" password="pwd"/>
  <user name="Fred" password="fredpwd"/>
  <user name="distuser1" password="pwd"/>
  <user name="distuser2" password="pwd"/>
</basicRegistry>
```

```
RACMAP ID(FRED) MAP USERDIDFILTER(NAME('Fred'))
  REGISTRY(NAME('*')) WITHLABEL('zCEE JWT FRED')
RACMAP ID(USER1) MAP USERDIDFILTER(NAME('distributed_User1'))
  REGISTRY(NAME('*')) WITHLABEL('zCEE JWT distributedUser1')
RACMAP ID(USER1) MAP USERDIDFILTER(NAME('distuser1'))
  REGISTRY(NAME('*')) WITHLABEL('zCEE JWT distuser1')
RACMAP ID(USER2) MAP USERDIDFILTER(NAME('distuser2'))
  REGISTRY(NAME('*')) WITHLABEL('zCEE JWT distuser2')
```



Tech/Tip: RACMAP Command Summary

```
RACMAP ID(USER1) MAP USERDIDFILTER(NAME('distuser1'))
    REGISTRY(NAME('*')) WITHLABEL('zCEE token user1')
RACMAP ID(USER1) MAP USERDIDFILTER(NAME('distribute_User1'))
    REGISTRY(NAME('zCEERealm')) WITHLABEL('zCEE user1')
RACMAP ID(USER1) MAP USERDIDFILTER(NAME('UID=user1,CN=User Name,OU=IBM ATG,O=IBM,C=US'))
    registry(name('*')) withlabel('USER X500 DN')
RACMAP ID(ATSUSER) MAP USERDIDFILTER(NAME('OU=IBM ATS,O=IBM,C=US'))
    registry(name('*')) withlabel('ATS USER')
RACMAP ID(IBMUSER) MAP USERDIDFILTER(NAME('O=IBM,C=US'))
    registry(name('*')) withlabel('IBM USER')
```

```
RACMAP ID(USER1) LISTMAP(LABEL('USER X500 DN'))

RACMAP ID(USER1) DELMAP (LABEL('zCEE distuser1'))

RACMAP QUERY USERDIDFILTER(NAME('USER1')) REGISTRY(NAME('*'))
```

```
RACMAP ID(USER1) LISTMAP
Label: zCEE token user1
Distributed Identity User Name Filter:
>distuser1<
Registry Name:
>*<

Label: zCEE user1
Distributed Identity User Name Filter:
>distribute_User1<
Registry Name:
>zCEERealm<

Label: USER X500 DN
Distributed Identity User Name Filter:
>UID=user1,CN=User Name,OU=IBM ATG,O=IBM,C=US<
Registry Name:
>*<
```



Liberty OpenID Client identity mapping configuration attributes

Decoded EDIT THE PAYLOAD AND SECRET

```
HEADER: ALGORITHM & TOKEN TYPE
{
  "kid": "kvjtqdlMjOTWiJrjOr73fu2MMt-FjiQrxU0YBzJLR4o",
  "alg": "RS256"
}

PAYLOAD: DATA
{
  "sub": "auser",
  "token_type": "Bearer",
  "scope": [
    "openid",
    "profile",
    "email"
  ],
  "azp": "rpSsl",
  "iss": "https://wg31.washington.ibm.com:26213/oidc/endpoint/OP",
  "aud": "myZcee",
  "exp": 1646761228,
  "iat": 1646760928,
  "realmName": "zCEERealm",
  "uniqueSecurityName": "auser"
}
```

```
<safRegistry id="saf" />
<safAuthorization racRouteLog="ASIS" />
<safCredentials unauthenticatedUser="WSGUEST"
  mapDistributedIdentities="true" ←
  profilePrefix="BBGZDFLT" />
```

Use distributed identity filters to map the distributed identities to SAF user IDs, using IDIDMAP resources and the RACMAP command.

```
<authFilter id="ATSAuthFilter">
  <requestUrl id="ATSDemoUrl"
    name="ATSRefererUri"
    matchType="contains"
    urlPattern="/cscvinc/employee|/db2/employee|/mqapi/loan"/>
</authFilter>
<openidConnectClient id="ATS"
  httpsRequired="true"
  authFilterRef="ATSAuthFilter"
  inboundPropagation="required"
  scope="openid profile email"
  audiences="myZcee"
  issuerIdentifier="https://wg31.washington.ibm.com:26213/oidc/endpoint/OP"
  mapIdentityToRegistryUser="false" ←
  signatureAlgorithm="RS256"
  userIdentityToCreateSubject="sub"
  trustAliasName="JWT-Signer-Certificate"
  trustStoreRef="jwtTrustStore"
  authnSessionDisabled="true"
  disableLtpaCookie="true">
</openidConnectClient>
<keyStore fileBased="false" id="jwtTrustStore"
  location="safkeyring:///JWT.KeyRing"
  password="password" readOnly="true" type="JCERACFKS"/>
```

Specifies whether to map the identity to a registry user. If this is set to false, then the user registry (SAF) is not used to create the user subject.

Liberty/zCEE OpenID Client identity mapping configuration attributes (JWK)



```
{  
    "kid": "574eafad-fcb5-412e-97a3-8100a1c1fa5b",  
    "alg": "RS256"  
}  
  
{  
    "sub": "mitchj",  
    "aud": "myZCEE",  
    "iss": "https://wg31.washington.ibm.com:26213/oidc/endpoint/OP",  
    "exp": 1610451176,  
    "iat": 1610451876  
}
```

```
<openidConnectClient  
    id="ATSJWK"  
    clientId="RS-JWT-ZCEE"  
    httpsRequired="true"  
    authFilterRef="jwkAuthFilter"  
    inboundPropagation="required"  
    signatureAlgorithm="RS256"  
    userIdentifier="sub"  
    mapIdentityToRegistryUser="true"  
    issuerIdentifier="https://wg31.washington.ibm.com:26213/oidc/endpoint/OP"  
    disableLtpaCookie="true"  
    audiences="myZcee"  
    tokenReuse="true"  
    jwkEndpointUrl="https://wg31.washington.ibm.com:26213/oidc/endpoint/OP/jwk"  
    jwkClientId="jwtClient"  
    jwkSecret="jwtSecret"/>  
</openidConnectClient>
```

JWT used in scenario – putting it all together

```
{  
  "alg": "RS256"  
}  
  
{  
  "sub": "Edward Johnson",  
  "token_type": "Bearer",  
  "azp": "rpSsl",  
  "iss": "https://wg31.washington.ibm.com:26213/oidc/endpoint/OPssl",  
  "aud": "myZcee",  
  "realmName": "zCEERealm",  
  "uniqueSecurityName": "Edward Johnson"  
}  
RSASHA256(base64UrlEncode(header) + base64UrlEncode(payload)
```

- The header contains an **alg** (algorithm) element value **RS256**
 - **RS256** (RSA Signature with SHA-256) is an asymmetric algorithm which uses a **public/private** key pair
 - **ES512** (Elliptic Curve Digital Signature Algorithm with SHA-512) [link for more info](#)
 - **HS256** (HMAC with SHA-256) is a symmetric algorithm with only one (**secret**) key
- The **iss** (issuer) claim identifies the principal that issued the JWT
- The **sub** (subject) claim **distuser** identifies the principal that is the subject of the JWT
- The **aud** (audience) claim **myZcee** identifies the recipients for which the JWT is intended

Configuring authentication with JWT



Liberty can perform user authentication with JWT using the support that is provided by the *openidConnectClient-1.0* feature. The **<openidConnectClient>** element is used to accept a JWT token as an authentication token

```
<openidConnectClient id="RPssl" inboundPropagation="required"
    signatureAlgorithm="RS256" trustAliasName="JWT-Signer"
    trustStoreRef="jwtTrustStore"
    userIdentityToCreateSubject="sub" mapIdentityToRegistryUser="false"
    issuerIdentifier="https://wg31.washington.ibm.com:26213/oidc/endpoint/OPssl"
    authnSessionDisabled="true" audiences="myZcee"/>
```

- ***inboundPropagation*** is set to required to allow z/OS Connect EE to use the received JWT as an authentication token
- ***signatureAlgorithm*** specifies the algorithm to be used to verify the JWT signature
- ***trustStoreRef*** specifies the name of the keystore element that defines the location of the validating certificate
- ***trustAliasName*** gives the alias or label of the certificate to be used for signature validation
- ***userIdentityToCreateSubject*** indicates the claim to use to create the user subject
- ***mapIdentityToRegistryUser*** indicates whether to map the retrieved identity to the registry user
- ***issuerIdentifier*** defines the expected issuer
- ***authnSessionDisabled*** indicates whether a WebSphere custom cookie should be generated for the session
- ***audiences*** defines a list of target audiences

Using authorization filters



Authentication filter can be used to filter criteria that are specified in the **authFilter** element to determine whether certain requests are processed by certain providers, such as OpenID Connect, for authentication.

```
<openidConnectClient id="RPssl" inboundPropagation="required"
    signatureAlgorithm="RS256" trustAliasName="JWT-Signer"
    trustStoreRef="jwtTrustStore"
    userIdentityToCreateSubject="sub" mapIdentityToRegistryUser= "true"
    issuerIdentifier="https://wg31.washington.ibm.com:26213/oidc/endpoint/OPssl"
    authnSessionDisabled="true" audiences="myZcee"
    authFilterRef="JwtAuthFilter" />
<openidConnectClient id="RPsslG" . . . authFilterRef= "API Gateway" />
<openidConnectClient id="RPsslURL" . . . authFilterRef= "URLFilter" />
<authFilter id="API Gateway">
    <remoteAddress id="ApiAddress" ip="10.7.1.*" matchType="equals" />
</authFilter>
<authFilter id="URLFilter">
    <requestUrl id="URL" urlPattern="/cscvinc/employee|/db2/employee|/mqapi/loan" />
    matchType="equals" /> </authFilter>
<authFilter id="JwtAuthFilter" >
    <requestHeader id="authHeader" name="Authorization" value="Bearer" matchType="contains" />
</authFilter>
```

Some alternative filter types

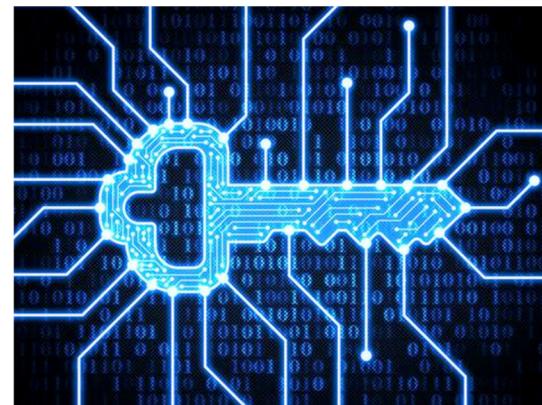
- A **remoteAddress** element is compared against the TCP/IP address of the client that sent the request.
- The **host** element is compared against the "Host" HTTP request header, which identifies the target host name of the request.
- The **requestUrl** element is compared against the URL that is used by the client application to make the request.

General security terms or considerations

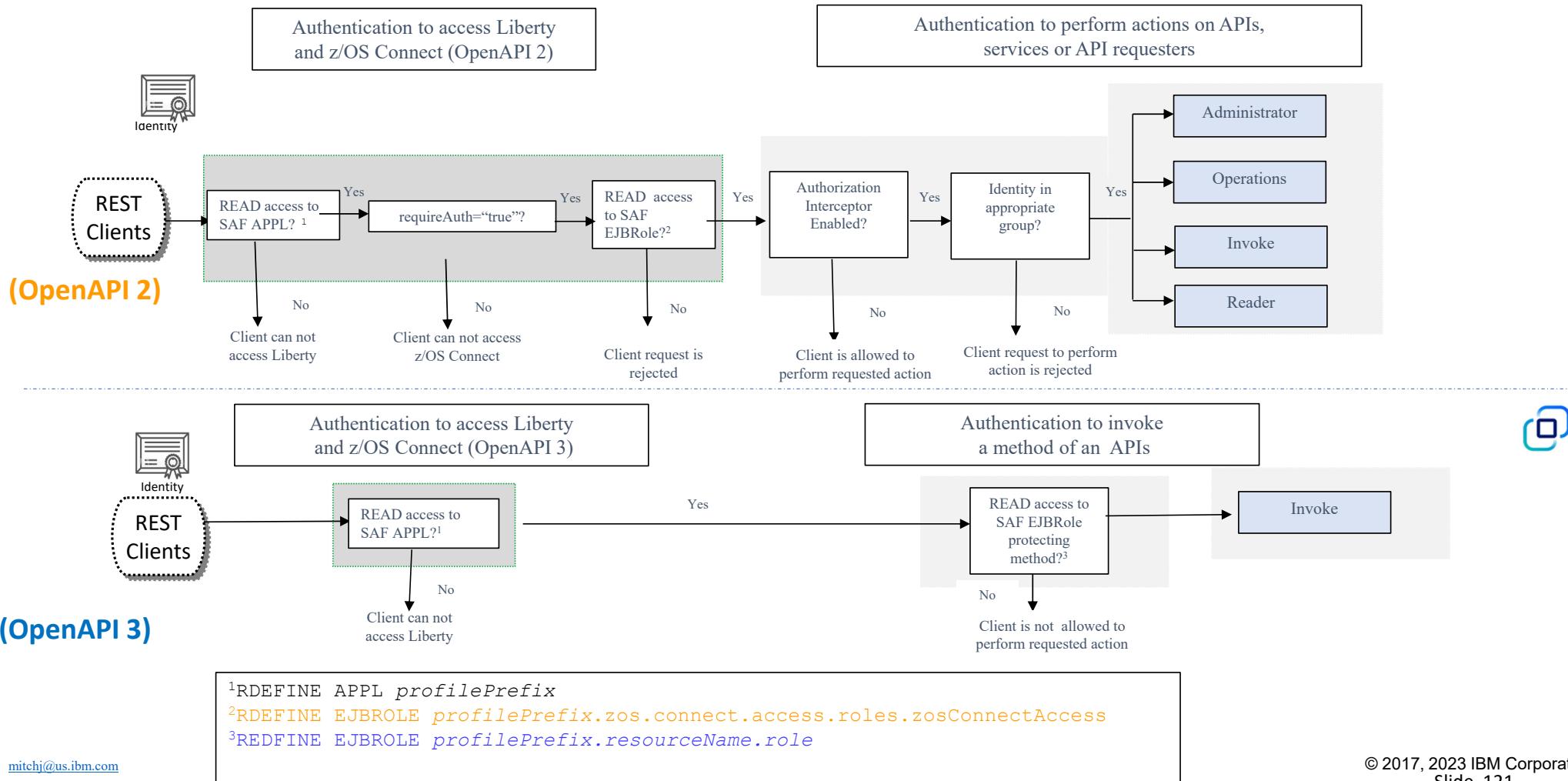
Security involves

- Identifying who or what is requesting access (**Authentication**)
 - Basic Authentication
 - Mutual Authentication using Transport Layer Security (TLS), formerly known as SSL
 - Third Party Tokens
- Ensuring that the message has not been altered in transit (**Data Integrity**) and ensuring the confidentiality of the message in transit (**Encryption**)
 - TLS (encrypting messages and using a digital signature)

- Controlling access (**Authorization**)
 - Is the authenticated identity authorized to access to z/OS Connect
 - Is the authenticated identity authorized to access a specific API, Services, etc.



Authorization security flow within z/OS Connect



z/OS Connect Authorization Functions (OpenAPI 2)



Operations - Ability to perform all z/OS Connect EE operations and actions except for function *Invoke*. The following operations/actions are allowed:

APIs:

- *To obtain a list of all APIs (GET).**
- For a specific API, get its details and API Swagger document (GET) and *deploy (POST)**, update (PUT), start(PUT), stop(PUT), and delete(DELETE) it.

Services:

- *To obtain a list of all services or statistics for all services (GET).**
- For a specific service, get its details, request and response schemas, statistics (GET) and *deploy(POST)**, update(PUT), start(PUT), stop(PUT), and delete(DELETE) it.

API Requesters:

- *To obtain a list of all API requesters (GET).**
- For a specific API requester, get its details (GET) and *deploy (POST)**, update(PUT), start(PUT), stop(PUT), and delete(DELETE) it.

*These APIs use either the POST or GET method to invoke the REST APIs whose URIs have no path parameter. Therefore, the name of the API, or service or API Requester is ignored. For authorization, only the default or global groups list can be used since no specific group list can be determined (for deployment, the name is embedded in the archive file).



z/OS Connect Authorization Levels (OpenAPI 2)

Reader - Ability for:

APIs:

- *To obtain a list of all APIs (GET) . **
- For a specific API, get its details and API Swagger document (GET).

Services:

- *To obtain a list of all services (GET). **
- For a specific service, get its details and request and response schemas (GET).

API Requesters:

- *To obtain a list of all API requesters (GET). **
- For a specific API requester, get its details (GET) .

Invoke - Ability to invoke user APIs, services and/or API requesters (POST,PUT,GET,DELETE,+).

Admin - All z/OS Connect EE actions are allowed, including all corresponding *Operations*, *Invoke*, and *Reader* actions configured for the same z/OS Connect resource.

*These APIs use either the POST or GET method to invoke the REST APIs whose URIs have no path parameter. Therefore, the name of the API, service or API Requester is not available. For authorization, only the default or global groups list since no specific group list can be determined (for deployment, the name is embedded in the archive file).

z/OS Connect RESTful Administrative APIs Security (OpenAPI 2)



z/OS Connect uses group security for controlling authorization for accessing APIs. There are sets of default global groups for functional roles are configured in a `zosConnectManager` configuration element as shown below:

```
<zosconnect_zosConnectManager  
    globalInterceptorsRef="interceptorList_g"  
    globalAdminGroup="SYSPGRP" globalOperationsGroup="GBLOPERS"  
    globalInvokeGroup="GBLINVKE" globalReaderGroup="GBLRDR"/>
```

There are four classes of groups available controlling z/OS Connect functions, administration, operations, invoking and reader in our server. An authenticated identity membership in one or more of these groups provides access to the corresponding function to that identity.

There is also a way to provide an alternative set of groups for functional roles for specific APIs, services, and API requesters in subordinate configuration elements in our server.

```
<zosConnectAPI name="cscvinc"  
    adminGroup="CSCADMIN" operationsGroup="CSCOPERS"  
    invokeGroup="CSCINVKE" readerGroup="CSCREADR"/>  
  
<service name="cscvincSelectService"  
    adminGroup="CSCADMIN" operationsGroup="CSCOPERS"  
    invokeGroup="CSCINVKE" readerGroup="CSCREADR"/>  
  
<apiRequester name="cscvinc_1.0.0"  
    adminGroup="CSCADMIN" operationsGroup="CSCOPERS"  
    invokeGroup="CSCINVKE" readerGroup="CSCREADR"/>
```



Interceptor - server XML example (OpenAPI 2)

```
<zosconnect_zosConnectManager  
    globalInterceptorsRef="interceptorList_g"  
    globalAdminGroup="SYSPGRP"  
    globalOperationsGroup="GBLOPERS"  
    globalInvokeGroup="GBLINVKE"  
    globalReaderGroup="GBLDRR"/>  
  
<zosconnect_authorizationInterceptor id="auth"/>  
<zosconnect_auditInterceptor id="audit"/>  
<zosconnect_zosConnectInterceptors id="interceptorList_g"  
    interceptorRef="auth"/>  
<zosconnect_zosConnectInterceptors id="interceptorList_a"  
    interceptorRef="auth,audit"/>  
  
<zosconnect_zosConnectAPIs>  
    <zosConnectAPI name="catalog"  
        runGlobalInterceptorsRef="true"  
        adminGroup="aapigrp1,aapigrp2"  
        operationsGroup="oapigrp1,oapigrp2"  
        invokeGroup="iapigrp1,oapigrp2"  
        readerGroup="rapigrp1,rapigrp2"/>  
</zosconnect_zosConnectAPIs>  
  
<zosconnect_apiRequesters>  
    <apiRequester name="cscvincapi_1.0.0"  
        runGlobalInterceptorsRef="false"  
        interceptorsRef="interceptorList_a"  
        adminGroup="aaprgrp1,aaprgrp2"  
        operationsGroup="oaprgrp1,oaprgrp2"  
        invokeGroup="iaprgrp1,oaprgrp2"  
        readerGroup="raprgrp1,raprgrp2"/>  
</zosconnect_apiRequesters>  
  
<zosconnect_services>  
    <service id="selectByEmployee" name="selectEmployee"  
        runGlobalInterceptorsRef="false"  
        interceptorsRef="interceptorList_a"  
        adminGroup="asrvgrp1,asrvgrp2"  
        operationsGroup="osrvgrp1,osrvgrp2"  
        invokeGroup="isrvgrp1,isrvgrp2"  
        readerGroup="rsrvrgrp1,rsrvgrp2"/>  
</zosconnect_services>
```

```
ADDGROUP SYSPGRP OMVS (AUTOGID) *  
ADDGROUP GBLINVKE OMVS (AUTOGID) *  
CONNECT FRED GROUP (SYSPGRP)  
CONNECT USER1 GROUP (GBLINVKE)
```

Global interceptor list – authorization interceptor only

Alternative interceptor list – authorization and audit interceptors

This avoids duplication of interceptors

Note that these are z/OS Connect configuration elements. Documented in the z/OS Connect KC

*RDEFINE FACILITY BPX.NEXT.USER APPLDATA('2001/201')

Tech/Tip: Server XML example – combining TLS/AUTH interceptor (OpenAPI 2)



```
<zosconnect_zosConnectManager  
    requireAuth="true"  
    requireSecure="true"  
    globalInterceptorsRef="interceptorList_g"  
    globalAdminGroup="SYSPGRP"  
    globalOperationsGroup="GBLOPERS"  
    globalInvokeGroup="GBLINVKE"  
    globalReaderGroup="GBLRDR"/>  
  
<zosconnect_authorizationInterceptor id="auth"/>  
<zosconnect_zosConnectInterceptors id="interceptorList_g"  
    interceptorRef="auth"/>  
  
<zosconnect_apiRequesters>  
    <apiRequester name="cscvincapi_1.0.0"  
        requireSecure="false"  
        invokeGroup="iaprgrp1"/>  
</zosconnect_apiRequesters>
```

Global TLS security and authentication are enabled.

TLS security is disabled for this API requester archive artifact. Avoiding the HTTP 302 REDIRECT error.

This configuration would allow a MVS batch job to authenticate to z/OS Connect and use HTTP for the protocol (when an AT-TLS outbound policy is not available). Only authorization identities which are members of groups identified as administrators or invokers would be authorized to invoke this API requester.

F BAQSTRT,ZCON,CLEARSAFCACHE

Example of z/OS Connect Authorization Levels (Open API 2) (this config has issues)



```
<zosconnect_zosConnectManager
    globalInterceptorsRef="interceptorList_g"
    globalAdminGroup="SYSPGRP" globalOperationsGroup="GBLOPERS"
    globalInvokeGroup="GBLINVKE" globalReaderGroup="GBLRDR"/>

<zosconnect_zosConnectAPIs>
    <zosConnectAPI name="cscvinc"
        adminGroup="CSCADMIN" operationsGroup="CSCOPERS"
        invokeGroup="CSCINVKE" readerGroup="CSCREADR"/>
    <zosConnectAPI name="db2employee"
        adminGroup="DB2ADMIN" operationsGroup="DB2OPERS"
        invokeGroup="DB2INVKE" readerGroup="DB2READR"/>
</zosconnect_zosConnectAPIs>

<zosconnect_services>
    <service name="cscvincSelectService"
        adminGroup="CSCADMIN" operationsGroup="CSCOPERS"
        invokeGroup="CSCINVKE" readerGroup="CSCREADR"/>
    <service name="selectEmployee"
        adminGroup="DB2ADMIN" operationsGroup="DB2OPERS"
        invokeGroup="DB2INVKE" readerGroup="DB2READR"/>
</zosconnect_services>

<zosconnect_apiRequesters>
    <apiRequester name="cscvincSelectService"
        adminGroup="CSCADMIN" operationsGroup="CSCOPERS"
        invokeGroup="CSCINVKE" readerGroup="CSCREADR"/>
    <apiRequester name="selectEmployee"
        adminGroup="DB2ADMIN" operationsGroup="DB2OPERS"
        invokeGroup="DB2INVKE" readerGroup="DB2READR"/>
</zosconnect_apiRequesters>
```

This works as you expect once the artifacts are deployed but:

- Only members of groups SYSPGRP, GBLOPERS or GBLRDR can connect to a z/OS server from the API toolkit (the tooling attempts a GET request for a list of all deployed services and APIs).
- Only members of groups SYSPGRP or GBLOPERS can deploy new z/OS Connect API, service or API requester artifacts (POST access for operations is not available until after the artifact is deployed)

Tech-Tip: When groups are specified for zosConnectAPI, service, or apiRequester configuration elements, the global groups are ignored for certain functions. Other functions, e.g., deploy new artifact, get a list or service statistics, only use the global group membership.

z/OS Connect Authorization Summary (OpenAPI 2)



- Members of groups SYSPGRP, GBLOPERS, DB2ADMIN or DB2OPERS can not manage (e.g., change, stop or delete) z/OS Connect artifacts *managed* by group CSCOPERS or CSCADMIN.
- Members of groups SYSPGRP, GBLOPERS, CSCADMIN or CSCOPERS can not manage (e.g., change, stop or delete) z/OS Connect artifacts *managed* by group DB2OPERS or DB2ADMIN.
- Only members of group CSCADMIN, CSCINV, DB2ADMIN or DB2INVKE can invoke the artifacts defined in the subordinate element:
 - Members of group CSCADMIN or CSCVINKE can invoke artifacts managed by CSCINVKE
 - Members of group DB2ADMIN or DB2INVKE can invoke artifacts managed by DB2INVKE
 - Members of groups SYSPGRP or GBLINVKE can not invoke any artifacts protected by these specific subordinate groups.
- Only members of groups SYSPGRP, GBLOPERS or GBLRDR can connect to a z/OS server from the API toolkit.
- Only members of groups SYSPGRP or GBLOPERS can deploy new z/OS Connect API, service or API requester artifacts.



Tech-Tip: Solution for z/OS Connect Authorization Levels (OpenAPI 2)

```
<zosconnect_zosConnectManager>
    globalInterceptorsRef="interceptorList_g"
    globalAdminGroup="SYSPGRP" globalOperationsGroup="GBLOPERS , CSCOPERS , DB2OPERS"
    globalInvokeGroup="GBLINVKE" globalReaderGroup="GBLRDR"/>

<zosconnect_zosConnectAPIs>
    <zosConnectAPI name="cscvinc" operationsGroup="CSCOPERS" invokeGroup="CSCINV"/>
    <zosConnectAPI name="db2employee" operationsGroup="DB2OPERS" invokeGroup="DB2INVKE"/>
</zosconnect_zosConnectAPIs>

<zosconnect_services>
    <service name="cscvincSelectService" operationsGroup="CSCOPERS" invokeGroup="CSCINV"/>
    <service name="selectEmployee" operationsGroup="DB2OPERS" invokeGroup="DB2INVKE"/>
</zosconnect_services>

<zosconnect_apiRequesters>
    <apiRequester name="cscvincSelectService" operationsGroup="CSCOPERS" invokeGroup="CSCINV"/>
    <apiRequester name="selectEmployee" operationsGroup="DB2OPERS" invokeGroup="DB2INVKE"/>
</zosconnect_apiRequesters>
```

- Now members of groups SYSPGRP, GBLOPERS, **CSCOPERS**, **DB2OPERS** and GBLRDR can connect to a z/OS server from the API toolkit.
- Members of groups SYSPGRP, GBLOPERS, **CSCOPERS**, and **DB2OPERS** can deploy new artifacts.
- Only members of group **CSCOPERS** and **DB2OPERS** can manage artifacts after they are deployed.



Tech-Tip: z/OS Toolkit and authorization status (OpenAPI 2)

Members of CSCOPERS and DB2OPERS can now connect to a server from the API Toolkit

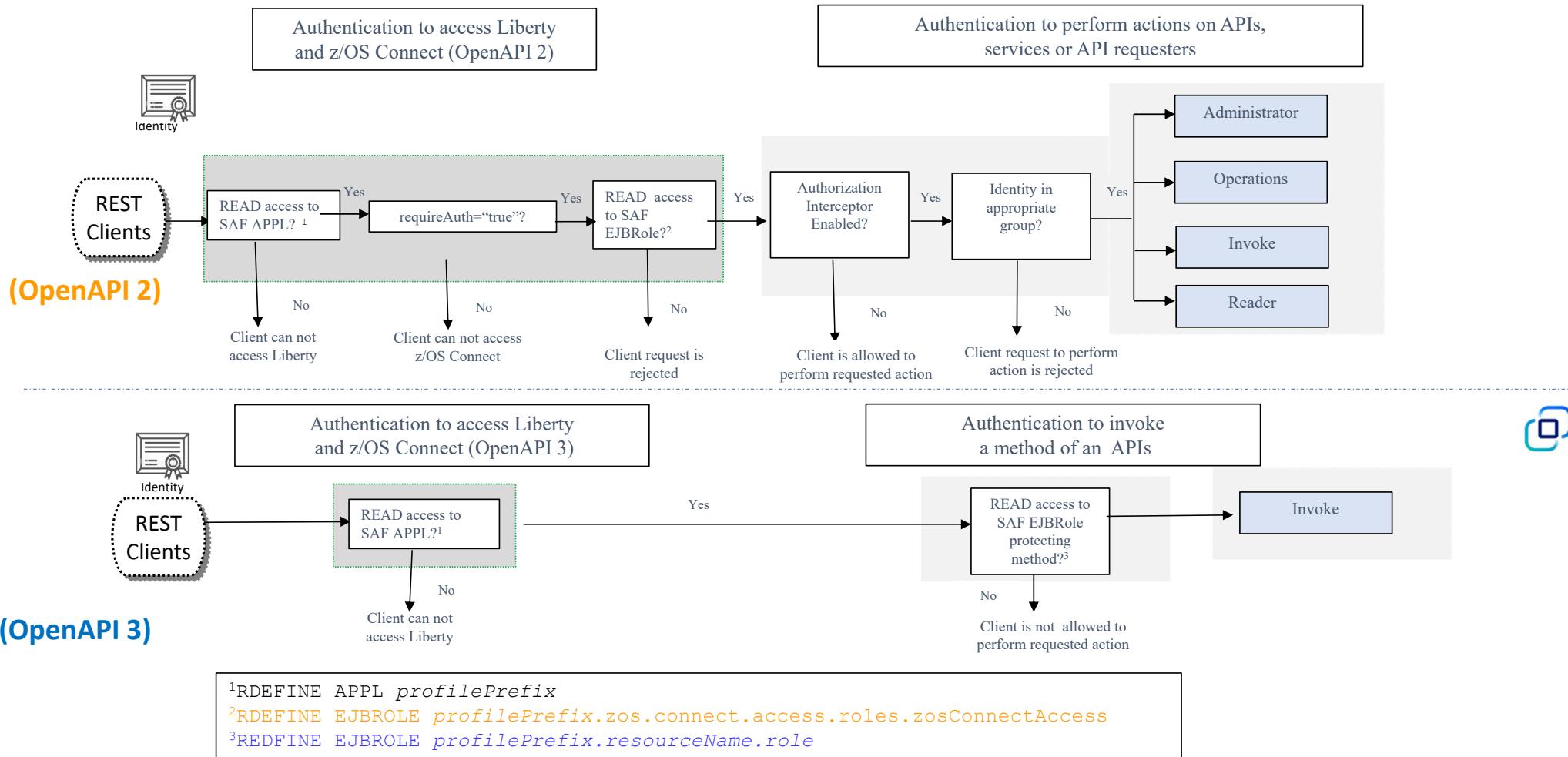
CSCOPERS

The screenshot shows the 'z/OS Connect EE Servers' interface with the 'Remote Systems' tab selected. Under the 'wg31:9443 (wg31.washington.ibm.com:9443)' node, the 'APIs (9)' section is expanded. It lists several services, with 'db2employee' highlighted by a red oval. The status of 'db2employee' is shown as '(Not Authorized)'. Other listed services include cscvinc, filemgr, imsPhoneBook, jvtlvpDemoApi, miniloancics, mqapi, phonebook, and restadmin, all marked as 'Started'.

DB2OPERS

The screenshot shows the same 'z/OS Connect EE Servers' interface for the DB2OPERS group. The 'APIs (9)' section for 'wg31:9443' is expanded, showing the same list of services as the CSCOPERS screenshot. However, the 'db2employee' service is now highlighted by a red oval, and its status is shown as '(Not Authorized)'. The other services (cscvinc, filemgr, imsPhoneBook, jvtlvpDemoApi, miniloancics, mqapi, phonebook, restadmin) are all marked as 'Started'.

Authorization security flow with z/OS Connect



¹RDEFINE APPL *profilePrefix*

²RDEFINE EJBROLE *profilePrefix.zos.connect.access.roles.zosConnectAccess*

³REDEFINE EJBROLE *profilePrefix.resourceName.role*



EJB roles for z/OS Connect (OpenAPI 3)

```
<safCredentials unauthenticatedUser="WSGUEST" profilePrefix="BBGZDFLT" />  
  
<webApplication id="catalogManager" name="catalogManager"  
location="${server.config.dir}/apps/api.war" contextRoot="/catalogManager" />  
  
<safRoleMapper profilePattern=%profilePrefix%.%resourceName%.%role%
```

```
openapi: 3.0.0  
 . . .  
servers:  
- url: /  
x-ibm-zcon-roles-allowed:  
- Manager  
 . . .  
paths:  
/items:  
  get:  
    operationId: itemsGet  
    . . .  
/items/{id}:  
  get:  
    . . .  
  operationId: itemsIdGet  
  x-ibm-zcon-roles-allowed:  
    - Staff  
/orders:  
  post:  
    . . .  
  operationId: ordersPost  
  x-ibm-zcon-roles-allowed:  
    - Staff
```

*From the OpenApi document, the value for %role% would be either **Manager** or **Staff**.*

So, the required SAF EJB roles to be defined would be:

- *BBGZDFLT.catalogManager.Manager*
- *BBGZDFLT.catalogManager.Staff*

*REDFINE EJBROLE BBGZDFLT.catalogManager.Manager
REDFINE EJBROLE BBGZDFLT.catalogManager.Staff*

Access to use the GET method to invoke `/items` would require read access to EJB role *BBGZDFLT.catalogManager.Manager*.

Access to use the GET method to invoke `/items/{id}` and the POST method to invoke `/orders` would require read access to EJB role *BBGZDFLT.catalogManager.Staff*.



Deploying multiple APIs in the same native server (OpenAPI 3)

```
<webApplication id="catalogManager" name="catalogManager"  
location="${server.config.dir}/apps/catalogManager.war" contextRoot="/catalogManager" />  
<webApplication id="db2API" name="db2API"  
location="${server.config.dir}/apps/employees.war" contextRoot="/db2" />  
<webApplication id="cicsAPI" name="cicsAPI"  
location="${server.config.dir}/apps/api.war" contextRoot="/cics" />
```

catalogManager.war

```
/META-INF/openapi.yaml  
openapi: 3.0.0  
...  
servers:  
- url: /
```

catalogManager.war

```
/META-INF/openapi.yaml  
openapi: 3.0.0  
...  
servers:  
- url: /catalogManager
```

employees.war

```
/META-INF/openapi.yaml  
openapi: 3.0.0  
...  
servers:  
- url: /
```

employees.war

```
/META-INF/openapi.yaml  
openapi: 3.0.0  
...  
servers:  
- url: /db2
```

cscvinc.war

```
/META-INF/openapi.yaml  
openapi: 3.0.0  
...  
servers:  
- url: /
```

cscvinc.war

```
/META-INF/openapi.yaml  
openapi: 3.0.0  
...  
servers:  
- url: /cics
```



Flowing identities to back-end z/OS systems



Basic authentication - Identity and Password

Server XML Configuration elements where basic authentication can be provided.

```
<connectionFactory id="imsTM"> containerAuthDataRef="IMScredentials">
<authData id="IMScredentials" user= "identity" password= "password"/>

<connectionFactory id="imsDB">
<properties.imsudbJLocal databaseName="DFSIVPA" user="identity" password="password"/>
</connectionFactory>

<zosconnect_cicsIpicConnection id="CICS" authDataRef="CICScredentials"/>
<zosconnect_authData id="CICScredentials" user= "identity" password= "password"/>

<zosconnect_zosConnectServiceRestClientConnection id="Db2" basicAuthRef="db2Auth"/>
<zosconnect_zosConnectServiceRestClientBasicAuth id="db2Auth"
    userName="identity" password="password"/>

<jmsQueueConnectionFactory jndiName="MQ">
    <properties.wasJms userName="identity" password="password" />
</jmsQueueConnectionFactory>
```

The value of the password can be encoded in the server XML configuration file. Using the **securityUtility** shipped with WebSphere Liberty Profile.



Using securityUtility to encrypt passwords

Best practice : use encryption for passwords instead of base64 encoding

- **securityUtility** – located in <wlp_install_dir>/wlp/bin Usage: securityUtility {encode|createSSLCertificate|help} [options]

- For encryption, use encode --key=encryption_key
 - Specifies the key to be used when encoding using AES encryption. This string is hashed to produce an encryption key that is used to encrypt and decrypt the password. The key can be provided to the server by defining the variable **wlp.password.encryption.key** whose value is the key. If this option is not provided, a default key is used.

```
./securityUtility encode --encoding=aes --key=myKey myPassWord
```

```
{aes}AHO0aXdiVD96u4oMRhoKeYH3U7aDqtFXTuHFBsO98Wlb
```

- Support was added at Liberty 22.0.0.1 for storing an AES password encryption key in a SAF key ring, see URL
<https://www.ibm.com/docs/en/was-liberty/zos?topic=slia-storing-aes-password-encryption-key-in-saf-key-ring>

```
./securityUtility encode --encoding=aes --keyring=safkeyring://JOHNSON/Liberty.KeyRing --keyringType=JCERACFKS  
--keyLabel="Johnson Client Cert" myPassWord
```

- Also supports 1-way hash encoding – for passwords in server.xml with basicRegistry

- For hash, use encode --encoding=hash

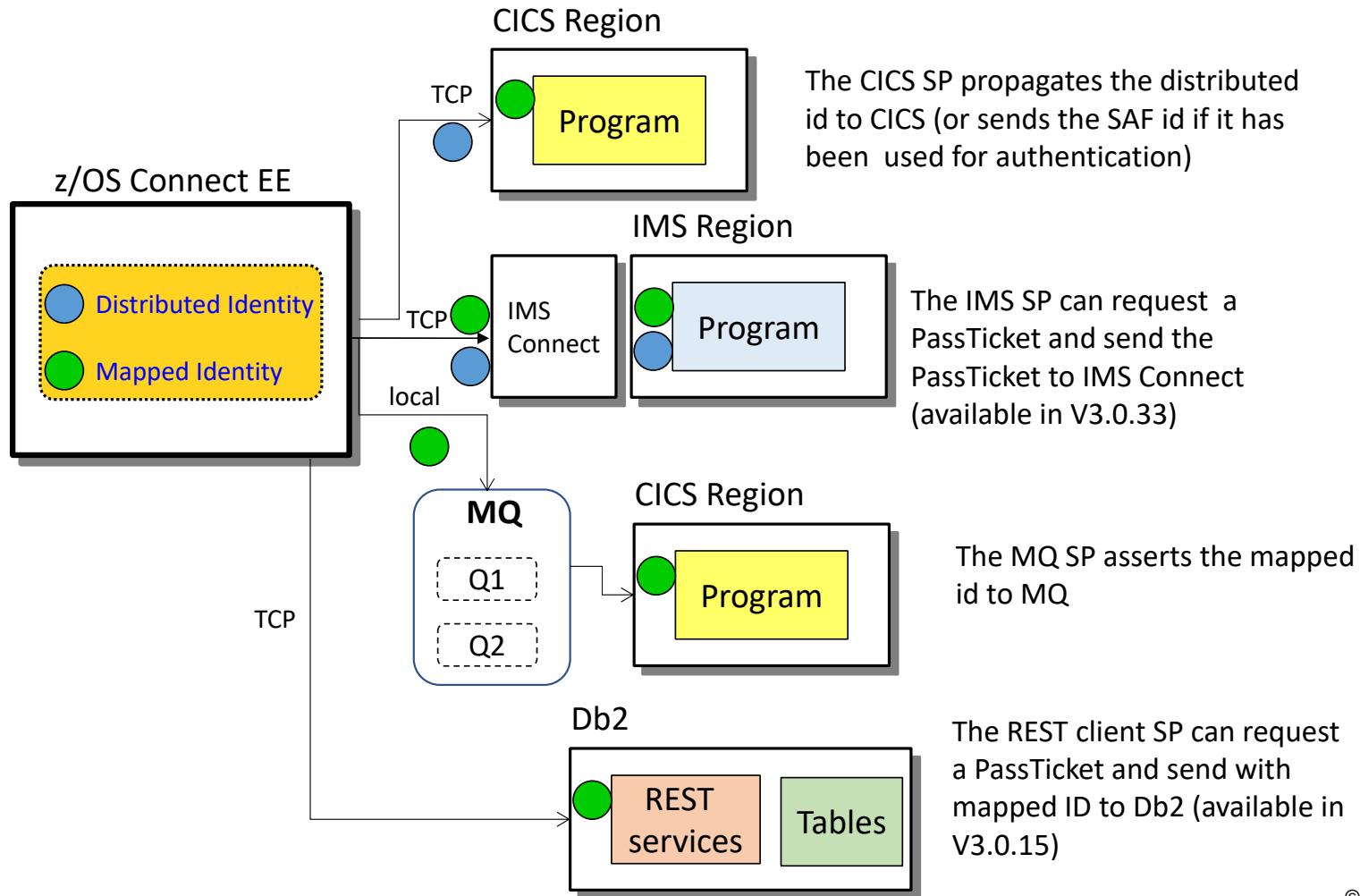
```
./securityUtility encode --encoding=hash XXXXXXXX
```

```
{hash}ATAAAAAIcqTmHn5qZahAAAAAIMjzy+hP8YFaIO6LiCreVe4etRLUS9a25eVuYtx6WKiv
```

See the WebSphere Application Server for z/OS Liberty *securityUtility* command at URL:

<https://www.ibm.com/docs/en/was-liberty/zos?topic=applications-securityutility-command>

Flowing an identity to a back-end subsystem





Flowing an identity to CICS

The zosconnect_cicsIpicConnection element is the key :

The screenshot shows the 'inquireSingle Service' configuration window. In the 'Required Configuration' section, 'Coded character set identifier (CCSID)' is set to 37 and 'Connection reference' is set to 'catalog'. In the 'Optional Configuration' section, 'Transaction ID' is present. The 'Configuration' tab is selected.

```
<zosconnect_cicsIpicConnection id="catalog"
    host="wg31.washington.ibm.com"
    zosConnectNetworkid="CSCVINC"
    zosConnectApplid="CSCVINC"
    port="1493"/>
```

The screenshot shows the 'WG31' TCP/IP configuration window. It displays various connection parameters such as host, port, protocol, and authentication settings. A red arrow points from the 'zosconnect_cicsIpicConnection' code to the 'host' field in the configuration window.

```
I_TCPIPS
RESULT - OVERTYPE TO MODIFY
Tcpiipservice(CSCVINC)
Openstatus( Open )
Port(01493)
Protocol(pic)
Ssltype(Nossl)
Type(FCSS)
Authenticate(Noauthentic)
Connections(000000)
Backlog( 01024 )
Maxdatalen( 000000 )
Urm( DFHISAPIP )
Privacy(Supported)
Ciphers(3538392F3233)
Host(ANY)
IpAddress(192.168.17.201)
Hosttype(Any)
Ipresolved(192.168.17.201)
+ Ipfamily(Ipv4family)
```

The screenshot shows the 'WG31' TCP/IP configuration window. It displays various connection parameters such as host, port, protocol, and authentication settings. A red arrow points from the 'zosconnect_cicsIpicConnection' code to the 'host' field in the configuration window.

```
I_TCPIPN
RESULT - OVERTYPE TO MODIFY
Ipconn(CSCVINC)
Applid(CSCVINC)
Networkid(CSCVINC)
Servstatus( Inservice )
Connstatus( Released )
Ssltype(Nossl)
Purgeinterval( 0 )
Hal(Notrequired)
Recvcount(001)
Sendcount(000)
Tcpiipservice(CSCVINC)
Port()
Host()
Hosttype()
Ipresolved(0.0.0.0)
Ipfamily(Unknown)
Pendstatus( Notpending )
+ Recovstatus( Norecovdata )
```

CICS IPCONN Resource



```
<zosconnect_cicsIpicConnection  
    id="cscvinc"  
    host="wg31.washington.ibm.com"  
    zosConnectApplid="ZOSAPPL"  
    zosConnectNetworkid="ZCNETID"  
    port="1491"/>
```

zosConnectApplid must match APPLID
in an IPCONN resource

zosConnectNetworkid must match
NETWORKID in an IPCONN resource

```
DEFINE IPCONN (ZOSCONN)  
GROUP (SYSPGRP)  
APPLID (ZCAPPL)  
NETWORKID (ZCNETID)  
TCPIPSERVICE (ZOSCONN)  
LINKAUTH (SECUSER | CERTUSER)  
USERAUTH (IDENTIFY)  
IDPROP (REQUIRED | OPTIONAL)
```

LINKAUTH Determines the user identity to be used for link security. The value is either **CERTUSER** or **SECUSER**. A value of **CERTUSER** sets the link identity to the identity associated with the client certificate received from the client endpoint (TLS mutual authentication is required). A value of **SECUSER** sets the link identity to the value of the *SECURITYNAME* attribute as defined in the IPCONN resource.

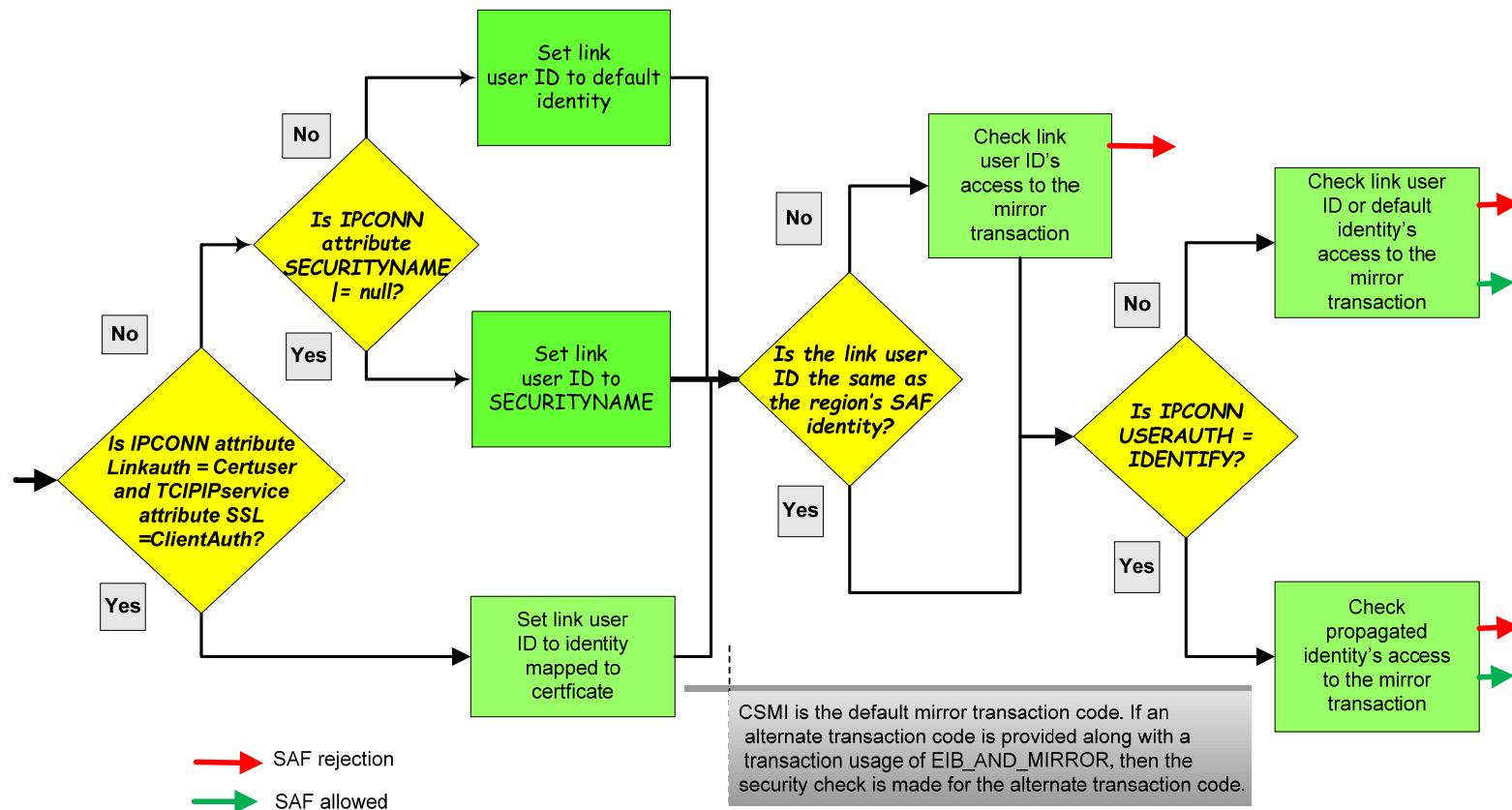
USERAUTH Identifies how the identity under which the attached transaction attach security will run. Since a password is not available, a value of **VERIFY** is not possible. A value of **LOCAL** means the current link identity is used. A value of **DEFAULTUSER** means the CICS default identity is used. For identity propagation purposes, the value of **USERAUTH** should be **IDENTIFY** (no password will be required) so the identity provided by the client is used for executing the attached transaction. TLS must be used if the client is in a different Sysplex.

IDPROP Determines whether the original distributed identity authenticated by the z/OS Connect server is also propagated to CICS in addition to the mapped identity used for z/OS Connect authorization checks. A value of **NOTALLOWED** does not propagate the original distributed identity. A value of **OPTIONAL** will propagate to CICS the original distributed identity, if available. A value of **REQUIRED** requires that the original distributed identity be propagated to CICS. TLS must be used if the client is in a different Sysplex.

CERTIFICATE Provides the label of the certificate connected to the CICS key ring to be used for server endpoint certificate during a TLS handshake.



Tech/Tip: CICS IPIC Security with USERAUTH(VERIFY)





Identity Propagation and CICS High Availability

Assume the service installed in a server files use the following *Connection reference* values:

- cscvinc
- catalog
- miniloan

If identity propagation is required for all connection, then the CICS IPCONN resources defined in the CICs that correspond to a `zosconnect_cicsIpicConnection` configuration elements must be dedicated to that z/OS Connect server and connection reference can not be reused.

Simplify administration by still sharing a common `cicsIpicConnection` XML configuration element by using variables and a bootstrap properties file or “variables” XML file

Server baqsvr1's bootstrap.properties

```
ipicPort=1491  
cicsHost=dvipa.washington.ibm.com  
serverPrefix=baqsvr1
```

Server baqsvr2's bootstrap.properties

```
cicsHost=dvipa.washington.ibm.com  
ipicPort=1491  
serverPrefix=baqsvr2
```

Server baqsvr3's bootstrap.properties

```
cicsHost=dvipa.washington.ibm.com  
ipicPort=1491  
serverPrefix=baqsvr3
```

ipicIDProp.xml

```
<zosconnect_cicsIpicConnection id="cscvinc"  
host="${cicsHost}"  
zosConnectNetworkid="${wlp.server.name}"  
zosConnectApplid="${wlp.server.name}"  
sharedPort="true" port="${ipicPort}"/>  
<zosconnect_cicsIpicConnection id="catalog"  
host="${cicsHost}"  
zosConnectNetworkid="${serverPrefix}C"  
zosConnectApplid="${serverPrefix}C"  
sharedPort="true" port="${ipicPort}"/>  
<zosconnect_cicsIpicConnection id="miniloan"  
host="${cicsHost}"  
zosConnectNetworkid="${serverPrefix}M"  
zosConnectApplid="${serverPrefix}M"  
sharedPort="true" port="${ipicPort}"/>
```

→ baqsvr1 or baqsvr2

→ baqsvr1C or baqsvr2C

→ baqsvr1M or baqsvr2M



CICS IPConn and TCPIPSERVICE resources for HA

CICS Specific TCPIPSERVICE - IPIC

```
TCpipservice : IPIC1
GROup       : SYSPGRP
Urm         ==> DFHISAIP
POrtnumber  ==> 01492
STatus      ==> Open
PROtocol    ==> IPic
TRansaction ==> CISS
Host        ==> ANY
Ipaddress   ==> ANY
SPeciftcp  ==>
```

CICS Generic TCPIPSERVICE - IPICG

```
TCpipservice : IPICG1
GROup       : SYSPGRP
Urm         ==> DFHISAIP
POrtnumber  ==> 01491
STatus      ==> Open
PROtocol    ==> IPic
TRansaction ==> CISS
Host        ==> ANY
Ipaddress   ==> ANY
SPeciftcp  ==> IPIC
```

A client connects first to the CICS region's generic port (1491) and then the CICS region redirects the client to the region's specific port (1492).

I IPConn ACQ

```
STATUS: RESULTS - OVERTYPE TO MODIFY
Ipc(BAQSVR1 ) App(BAQSVR1) Net(BAQSVR1) Ins Acq Nos
          Rece(001) Sen(000) Tcp(IPIC)
Ipc(BAQSVR1C) App(BAQSVR1C) Net(BAQSVR1C) Ins Acq Nos
          Rece(001) Sen(000) Tcp(IPIC)
Ipc(BAQSVR1M) App(BAQSVR1M) Net(BAQSVR1M) Ins Acq Nos
          Rece(001) Sen(000) Tcp(IPIC)
Ipc(BAQSVR2 ) App(BAQSVR2) Net(BAQSVR2) Ins Acq Nos
          Rece(001) Sen(000) Tcp(IPIC)
Ipc(BAQSVR2C) App(BAQSVR2C) Net(BAQSVR2C) Ins Acq Nos
          Rece(001) Sen(000) Tcp(IPIC)
Ipc(BAQSVR2M) App(BAQSVR2M) Net(BAQSVR2M) Ins Acq Nos
          Rece(001) Sen(000) Tcp(IPIC)
```

Number of IPConn resources equals the number of zCEE server times the number of unique connection references

¹CICS requires the specific TCPIPSERVICE be installed before the corresponding generic TCPIPSERVICE resource. TCPIPServices are installed in alphabetically order, so the name of specific service must be alphabetically prior to the name of the generic TCPIPSERVICE.

CICS IPIC connection processing for high availability load balancing*



If the *reconnectInterval* attribute is set, at the specified time interval, a check is made to see if a new connection attempt should be attempted. A new connection is established if the current connection properties are not the preferred connection properties:

- If *reconnectInterval*, *preferredSpecificHost* and *preferredSpecificPort* are not set,
 - New connection attempts are disabled (this is the default behavior).
- If *reconnectInterval* is set and *preferredSpecificHost* and *preferredSpecificPort* are not set,
 - A new connection is attempted at the interval specified by the *reconnectInterval* time. Use this to enable regular connection rebalancing.
- If *reconnectInterval* and *preferredSpecificPort* are set and *preferredSpecificHost* is not set,
 - A new connection is attempted at the expiration time interval and if the current connected port in use does not match the preferred port
 - Relevant when shared port is for a single LPAR
 - Specific CICS region is preferred
- If *reconnectInterval* and *preferredSpecificHost* are set and *preferredSpecificPort* is not set
 - A new connection is attempted at the expiration time interval and if the current host in use does not match the preferred port
 - Relevant when shared port is across Sysplex
 - Any CICS region on a specific LPAR is preferred
- If *reconnectInterval*, *preferredSpecificHost* and *preferredSpecificPort* are all set
 - A new connection is attempted at the expiration time interval time and if both the current host and port in use do not match the preferred host and port
 - Relevant when shared port is on a single LPAR or across a Sysplex
 - Specific CICS region is preferred.

When the reconnection attempt results in a new connection to a CICS region, new requests are sent over the new connection. Previous connections will continue and when all requests have completed processing, the previous or old connection will be closed.

Server XML - Accessing a Db2 REST service (OpenAPI 2)



*selectEmployee Service X

Service Project Editor: Configuration

Required Configuration

Enter the required configuration for this service.

Connection reference: db2conn

Definition Configuration

DSNL004I -DSN2 DDF START

COMPLETE

LOCATION DSN2LOC

LU

USIBMWZ.DSN2APPL

GENERICLU -NONE

DOMAIN

WG31.WASHINGTON.IBM.COM

TCPPORT 2446

SECPORT 2445

RESPORT 2447

```
<zosconnect_zosConnectServiceRestClientConnection id="db2conn"  
host="wg31.washington.ibm.com"  
port="2446"  
basicAuthRef="dsn2Auth" />  
  
<zosconnect_zosConnectServiceRestClientBasicAuth id="dsn2Auth"  
appName="DSN2APPL"/>
```



Server XML - Accessing a Db2 REST service (OpenAPI 3)

The screenshot shows the IBM z/OS Connect Designer interface. A flow diagram is displayed with the following components:

- A "Request" node (labeled "getEmployee-V1") connected to a "DB2" node.
- The "DB2" node has a label "getEmployee-V1" and a description "Get the details of all employees".
- An arrow points from the DB2 node to a "Responses" box.
- The "Responses" box contains a condition: "If commarea.DFH0XCP1.CA-RETURN-CODE e..." followed by a "200 OK" response.
- Below the flow diagram, the "getEmployee-V1" asset details are shown, including:
 - Name: getEmployee-V1
 - Type: Db2 native REST service
 - Description: Get the details of all employees
 - Db2 native REST service name: getEmployee
 - Collection ID: SYSIBMSERVICE
 - Db2 native REST service description: Get the details of all employees
- A red circle highlights the "Connection reference db2Conn" field under the "Db2 native REST service" section.

The connection references identifies a `zosconnect_db2Connection` configuration element. Which provides the connection details to a DB2 DDF task.

```
<featureManager>
  <feature>zosconnect:db2-1.0</feature>
</featureManager>

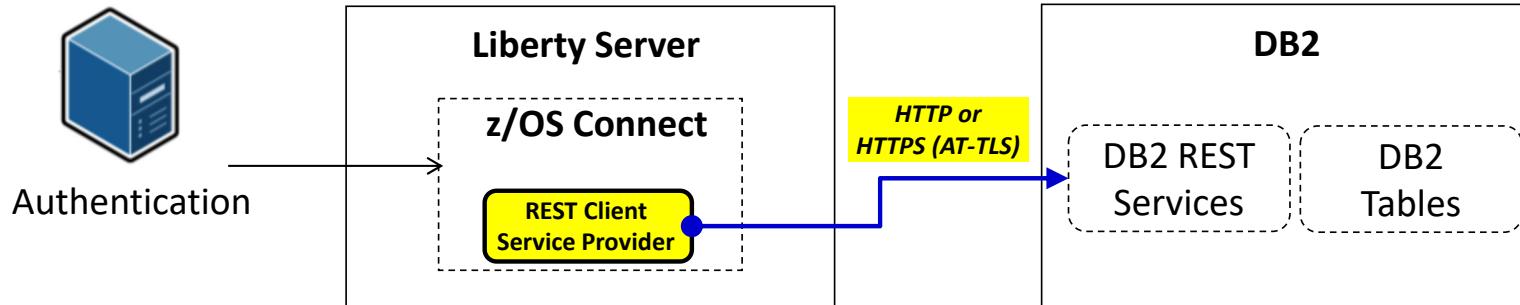
<zosconnect_credential user="${DB2_USERNAME}"
  password="${DB2_PASSWORD}" id="commonCredentials" />

<zosconnect_db2Connection id="db2Conn" host="${DB2_HOST}"
  port="${DB2_PORT}" credentialRef="commonCredentials" />
```

```
DSNL004I -DSN2 DDF START COMPLETE
LOCATION DSN2LOC
LU USIBMWZ.DSN2APPL
GENERICLU -NONE
DOMAIN WG31.WASHINGTON.IBM.COM
TCPPORT 2446
SECPORT 2445
RESPORT 2447
```

Define connections to Db2 using variables defined in bootstrap.properties file

Flowing the identity for the REST client SP (Db2)



```
<zosconnect_zosConnectServiceRestClientConnection id="Db2Conn"  
host="wg31.washington.ibm.com"  
port="2446"  
basicAuthRef="dsn2Auth" />  
<zosconnect_zosConnectServiceRestClientBasicAuth id="dsn2Auth"  
userName="USER1"  
password="USER1"/>
```

Authentication options:

1. User ID / password
2. TLS Client Certificate (JSSE)
3. PassTicket support

Specify a user identity and password to be used in the HTTP header with the Db2 REST Service

```
<zosconnect_zosConnectServiceRestClientConnection id="Db2Conn"  
host="wg31.washington.ibm.com"  
port="2446"  
basicAuthRef="dsn2Auth" />  
  
<zosconnect_zosConnectServiceRestClientBasicAuth id="dsn2Auth"  
appName="DSN2APPL"/>
```

z/OS Connect requests a PassTicket from RACF

PassTickets and Db2

- ☐ Basic authentication Db2 using a PassTicket depends on the Db2 configuration.

```
DSNL080I -DSN2 DSNLTDDF DISPLAY DDF REPORT FOLLOWS:
```

```
DSNL081I STATUS=STARTD
DSNL082I LOCATION      LUNAME          GENERICCLU
DSNL083I DSN2LOC       USIBMWZ.DSN2APPL  USIBMWZ.DSNOAPPL
DSNL084I TCPPORT=2446   SECPORT=2445    RESPORT=2447  IPNAME==NONE
DSNL085I IPADDR=:192.168.17.201
DSNL086I SQL   DOMAIN=WG31.WASHINGTON.IBM.COM
DSNL105I CURRENT DDF OPTIONS ARE:
DSNL106I PKGREL = COMMIT
DSNL106I SESSIDLE = 001440
DSNL099I DSNLTDDF DISPLAY DDF REPORT COMPLETE
```

```
RDEFINE PTKTDATA DSN2APPL SSIGNON(0123456789ABCDEF)
APPLDATA('NO REPLAY PROTECTION') UACC(NONE)
```

```
RDEFINE PTKTDATA IRRPTAUTH.DSN2APPL.* UACC(NONE)
PERMIT IRRPTAUTH.DSN2APPL.* ID(LIBSERV) CLASS(PTKTDATA)
ACCESS(UPDATE)
```

```
DSNL080I -DSNC DSNLTDDF DISPLAY DDF REPORT FOLLOWS:
```

```
DSNL081I STATUS=STARTD
DSNL082I LOCATION      LUNAME          GENERICCLU
DSNL083I DSN2LOC       -NONE          -NONE
DSNL084I TCPPORT=2446   SECPORT=2445    RESPORT=2447  IPNAME=DB2IPNM
DSNL085I IPADDR=:192.168.17.252
DSNL086I SQL   DOMAIN=WG31.WASHINGTON.IBM.COM
DSNL086I RESYNC DOMAIN=WG31.WASHINGTON.IBM.COM
DSNL089I MEMBER IPADDR=:192.168.17.252
DSNL105I CURRENT DDF OPTIONS ARE:
DSNL106I PKGREL = COMMIT
DSNL106I SESSIDLE = 001440
DSNL099I DSNLTDDF DISPLAY DDF REPORT COMPLETE
```

Which value should be used for *applName* is determined for use in RACF resources is determined as shown below.

- ☐ If *GENERICLU* is defined, use the second part of *GENERICLU* for *applName*, e.g., ***DSN0APPL***
- ☐ If *GENERICLU* is not defined, use the second part of *LUNAME* for *applName*, e.g., ***DSN2APPL***
- ☐ If neither *GENERICLU* or *LUNAME* is defined, use the value of the *IPNAME* for *applName*, e.g., ***DB2IPNM***



Tech/Tip: Db2 REST Security

- ❑ Access to Db2 REST services requires READ access to the Db2 subsystem DSNR REST resource. i.e., permit READ access to this resource to the identity in question, for example

```
PERMIT DSN2.REST CLASS(DSNR) ID(USER2) ACC(READ) where DSN2 is the Db2 subsystem ID  
SETROPTS RACLIST(DSNR) REFRESH
```

- ❑ Db2 package access is also required. If a user is not able to display a valid Db2 REST services in the z/OS Connect Db2 services development tooling or by using a **POST** to the Db2 provided REST interface URL of <http://wg31.washington.ibm.com:2446/services/DB2ServiceDiscover>, then they may not have sufficient access to the package containing the service.

For example, if service *zCEEService.selectEmployee* is defined to Db2 but not visible in the z/OS Connect tooling or if a **GET** request to URL <http://wg31.washington.ibm.com:2446/services/zCEEService/selectEmployee> fails with message:

```
{  
  "StatusCode": 500,  
  "StatusDescription": "Service zCEEService.selectEmployee discovery failed due to  
  SQLCODE=-551 SQLSTATE=42501, USER2 DOES NOT HAVE THE PRIVILEGE TO PERFORM OPERATION EXECUTE  
  PACKAGE ON OBJECT zCEEService.selectEmployee. Error Location:DSNLJACC:35"  
}
```

The user needs to be granted execute authority on package *zCEEService.selectEmployee* with command:

```
GRANT EXECUTE ON PACKAGE "zCEEService"."selectEmployee" TO USER2  or  
GRANT EXECUTE ON PACKAGE "zCEEService".*" TO USER2
```



Server XML – Accessing an IMS Transaction using OTMA (OpenAPI 2)

ivtnoService Service Configuration

Required Configuration

Enter the required configuration for this service.

Connection profile: **IMSCONN**

Interaction profile: **IMSINTER**

Optional Configuration

Enter the optional configuration for this service.

IMS destination override:

Program name:

Overview Configuration

IMS Connect HWSCFG

```
HWS=(ID=IMS14HWS,XIBAREA=100,RACF=Y,RRS=N)
TCPIP=(HOSTNAME=TCPIP,PORTID=(4000,LOCAL),RACFID=JOHNSON,TIMEOUT=5000)
DATASTORE=(GROUP=OTMAGRP,ID=IVP1, MEMBER=HWSMEM, T MEMBER=OTMAMEM)
IMSPLEX=(MEMBER=IMS14HWS, T MEMBER=PLEX1)
ODACCESS=(ODBMAUTOCONN=Y,
DRDAPORT=(ID=5555,PORTTMOT=6000), ODBMTMOT=6000)
```

connections/ims-connection.xml#

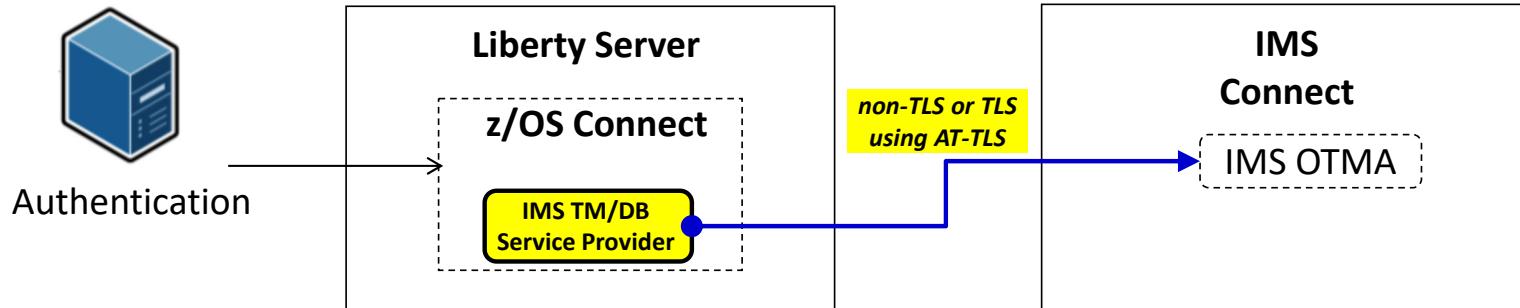
```
<server>
<imsmobile_imsConnection comment="" connectionFactoryRef="CF1" connectionTimeout="-1" connectionType="IMSCONNECT" id="IMSCONN"/>
<connectionFactory containerAuthDataRef="Connection1_Auth" id="CF1">
    <properties.gmoa hostName="wg31.washington.ibm.com" portNumber="4000"/>
</connectionFactory>

<authData id="Connection1_Auth" password="encryptedPassword1" user="userName1"/>
</server>
```

interactions/ims-interactions.xml#

```
<server>
<imsmobile_interaction comment="" commitMode="1" id="IMSINTER" imsConnectCodepage="Cp1047" imsConnectTimeout="0"
    imsDatastoreName="IVP1" interactionTimeout="-1" ltermOverrideName="" syncLevel="0"/>
</server>
```

Flowing an identity to IMS Connect (TM)



```
HWS=(ID=IMS15HWS,XIBAREA=100,RACF=Y,RRS=Y)  
TCPIP=(HOSTNAME=TCPIP,PORTID=(4000,LOCAL),RACFID=JOHNSON,TIMEOUT=5000)  
DATASTORE=(GROUP=OTMAGRP,ID=IVP1, MEMBER=HWSMEM, DRU=HWSYDRU0,  
TMEMBER=OTMAMEM,APPL=IMSTMAPP)
```

Authentication options:
1. User ID / password
2. PassTicket support

```
<connectionFactory containerAuthDataRef="Connection1_Auth" id="IVP1">  
<properties.gmoa hostName="wg31.washington.ibm.com" portNumber="4000"/>  
</connectionFactory>  
<authData id="Connection1_Auth" user="USER1" password="{xor}GhIPEExAGDwg="/>
```

Specify a user identity and password to be used in the request to IMS Connect

```
<connectionFactory containerAuthDataRef="Connection1_Auth" id="IVP1">  
<properties.gmoa hostName="wg31.washington.ibm.com" portNumber="4000"  
applicationName="IMSTMAPP"/>  
</connectionFactory>
```

Request a PassTicket
And use it in the request to IMS Connect

PassTickets and IMS

- ❑ Basic authentication to IMS Connect using a PassTicket depends on the APPL parameters configured in IMS Connect.

```
HWS=(ID=IMS15HWS,XIBAREA=100,RACF=Y,RRS=Y)
TCPIP=(HOSTNAME=TCPIP,PORTID=(4000,LOCAL),RACFID=JOHNSON,TIMEOUT=5000)
DATASTORE=(GROUP=OTMAGRP, ID=IVP1, MEMBER=HWSMEM, DRU=HWSYDRU0,
TMEMBER=OTMAMEM,APPL=IMSTMPL)
ODACCESS=(DBMAUTOCONN=Y, IMSPLEX=(MEMBER=IMS15HWS, TMEMBER=PLEX1),
DRDAPORT=(ID=5555, PORTMOT=6000), ODBMTMOT=6000,APPL=IMSDBAPL)
```

```
RDEFINE PTKTDATA IMSTMPL SSIGNON(0123456789ABCDEF) APPLDATA('NO REPLAY PROTECTION') UACC(NONE)
```

```
RDEFINE PTKTDATA IRRPTAAUTH.IMSTMPL.* UACC(NONE)
```

```
PERMIT IRRPTAAUTH.IMSTMPL.* ID(LIBSERV) CLASS(PTKTDATA) ACCESS(UPDATE)
```

```
RDEFINE PTKTDATA IMSDBAPL SSIGNON(0123456789ABCDEF) APPLDATA('NO REPLAY PROTECTION') UACC(NONE)
```

```
RDEFINE PTKTDATA IRRPTAAUTH.IMSDBAPL.* UACC(NONE)
```

```
PERMIT IRRPTAAUTH.IMSDBAPL.* ID(LIBSERV) CLASS(PTKTDATA) ACCESS(UPDATE)
```



Server XML – Accessing an IMS Database using ODBA (OpenAPI 2)

Service Project Editor: Configuration

Required Configuration

Enter the required configuration for this service.

Connection profile: DFSIVPACConn

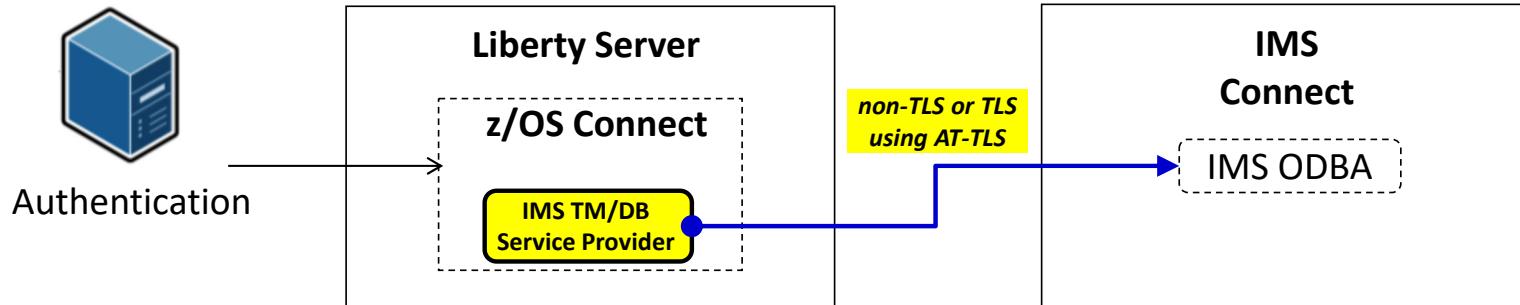
ConnectionFactory

```
<connectionFactory id="DFSIVPACConn">
<properties.imsudbJLocal
  databaseName="DFSIVPA"
  datastoreName="IVP1"
  datastoreServer="wg31.washington.ibm.com"
  driverType="4"
  portNumber="5555"
  user="USER1"
  password="password"
  flattenTables="True"/>
</connectionFactory>
```

IMS Connect HWSCFG

```
HWS=(ID=IMS14HWS,XIBAREA=100,RACE=N,RRS=N)
TCPIP=(HOSTNAME=TCPIP,PORTID=(4000,LOCAL),RACFID=JOHNSON,TIMEOUT=5000)
DATASTORE=(GROUP=OTMAGRP,ID=IVP1, MEMBER=HWSMEM, TMEMBER=OTMAMEM)
IMSPLEX=(MEMBER=IMS14HWS, TMEMBER=PLEX1)
ODACCESS=(ODBMAUTOCONN=Y,
DRDAPORT=(ID=5555,PORTTMOT=6000),ODBMTMOT=6000)
```

Flowing an identity to IMS Connect (DB)



```
HWS=(ID=IMS15HWS,XIBAREA=100,RACF=Y,RRS=Y)
TCPIP=(HOSTNAME=TCPIP,PORTID=(4000,LOCAL),RACFID=JOHNSON,TIMEOUT=5000)
ODACCESS=(ODBMAUTOCONN=Y,IMSPLEX=(MEMBER=IMS15HWS,TMEMBER=PLEX1),
DRDAPORT=(ID=5555,PORTTMOT=6000),ODBMTMOT=6000,APPL=IMSDBAPL)
```

- Authentication options:**
1. User ID / password
 2. PassTicket support

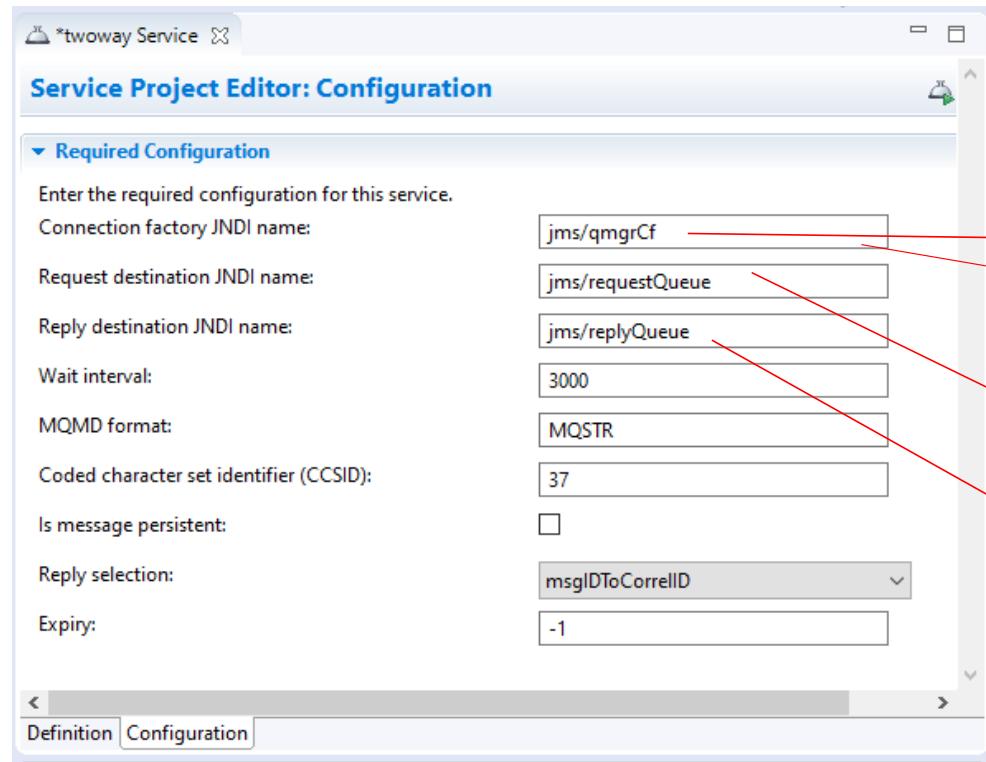
```
<connectionFactory id="DFSIVPACConn"> <properties.imsudbJLocal
databaseName="DFSIVPA" datastoreName="IVP1" portNumber="5555"
driverType="4" datastoreServer="wg31.washington.ibm.com" flattenTables="True"
user="USER1" password="USER1" />
</connectionFactory>
```

Specify a user identity and password to be used in the request to IMS Connect

```
<connectionFactory id="DFSIVPACConn"> <properties.imsudbJLocal
databaseName="DFSIVPA" datastoreName="IVP1" portNumber="5555"
datastoreServer="wg31.washington.ibm.com" driverType="4" flattenTables="True"
applicationName="IMSDBAPL" />
</connectionFactory>
```

Request a PassTicket
And use it in the request to IMS Connect

Server XML - Using JMS to access MQ (OpenAPI 2)



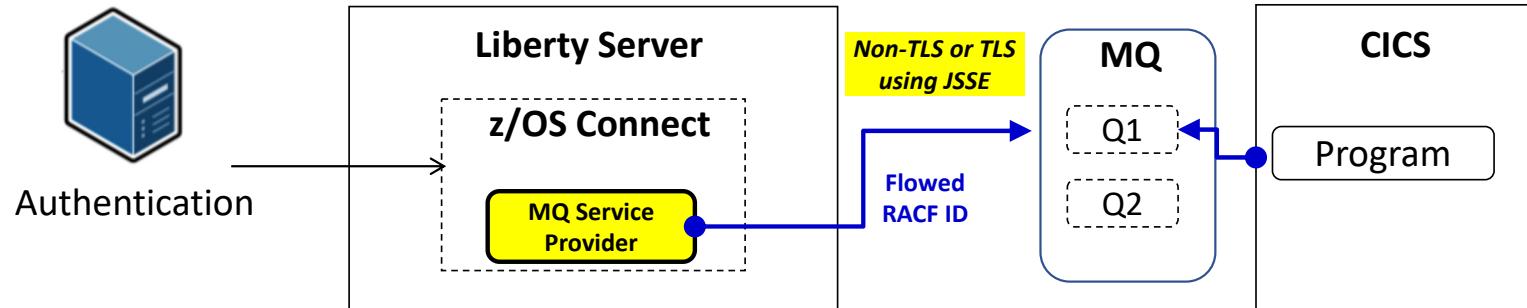
The screenshot shows the 'mq.xml' configuration file in 'Source' mode. Red arrows point from the Service Project Editor fields to the corresponding XML elements:

- A red arrow points from the 'jms/qmgrCf' field to the `jmsConnectionFactory id="qmgrCf" jndiName="jms/qmgrCf"` element.
- A red arrow points from the 'jms/requestQueue' field to the `jmsQueue id="requestQueue" jndiName="jms/request"` element.
- A red arrow points from the 'jms/replyQueue' field to the `jmsQueue id="replyQueue" jndiName="jms/replyQueue"` element.

```
2 <featureManager>
3   <feature>zosconnect:mqService-1.0</feature>
4 </featureManager>
5
6 <variable name="wmqJmsClient.rar.location"
7   value="/usr/lpp/mqm/V9R1M1/java/lib/jca/wmq.jmsra.rar"/>
8 <wmqJmsClient nativeLibraryPath="/usr/lpp/mqm/V9R1M1/java/lib"/>
9
10 <connectionManager id="ConMgr1" maxPoolSize="5"/>
11
12 <jmsConnectionFactory id="qmgrCf" jndiName="jms/qmgrCf">
13   connectionManagerRef="ConMgr1">
14   <properties.wmqJMS transportType="BINDINGS"
15     queueManager="QMZ1" />
16 </jmsConnectionFactory>
17
18 <jmsConnectionFactory id="qmgrCf2" jndiName="jms/qmgrCf2">
19   connectionManagerRef="ConMgr1">
20   <properties.wmqJMS transportType="CLIENT"
21     queueManager="ZMQ1"
22     channel="LIBERTY.DEF.SVRCONN"
23     hostName="wg31.washington.ibm.com"
24     port="1422" />
25 </jmsConnectionFactory>
26
27 <jmsQueue id="q1" jndiName="jms/default">
28   <properties.wmqJMS
29     baseQueueName="ZCONN2.DEFAULT.MQZCEE.QUEUE"
30     CCSID="37"/>
31 </jmsQueue>
32
33 <jmsQueue id="requestQueue" jndiName="jms/request">
34   <properties.wmqJMS
35     baseQueueName="ZCONN2.TRIGGER.REQUEST"
36     targetClient="MQ"
37     CCSID="37"/>
38 </jmsQueue>
39
40 <jmsQueue id="replyQueue" jndiName="jms/replyQueue">
41   <properties.wmqJMS
42     baseQueueName="ZCONN2.TRIGGER.RESPONSE"
43     targetClient="MQ"
44     CCSID="37"/>
45 </jmsQueue>
46
47
```



Flowing a user ID with MQ service provider



Set **useCallerPrincipal=true** to flow the authenticated RACF user ID

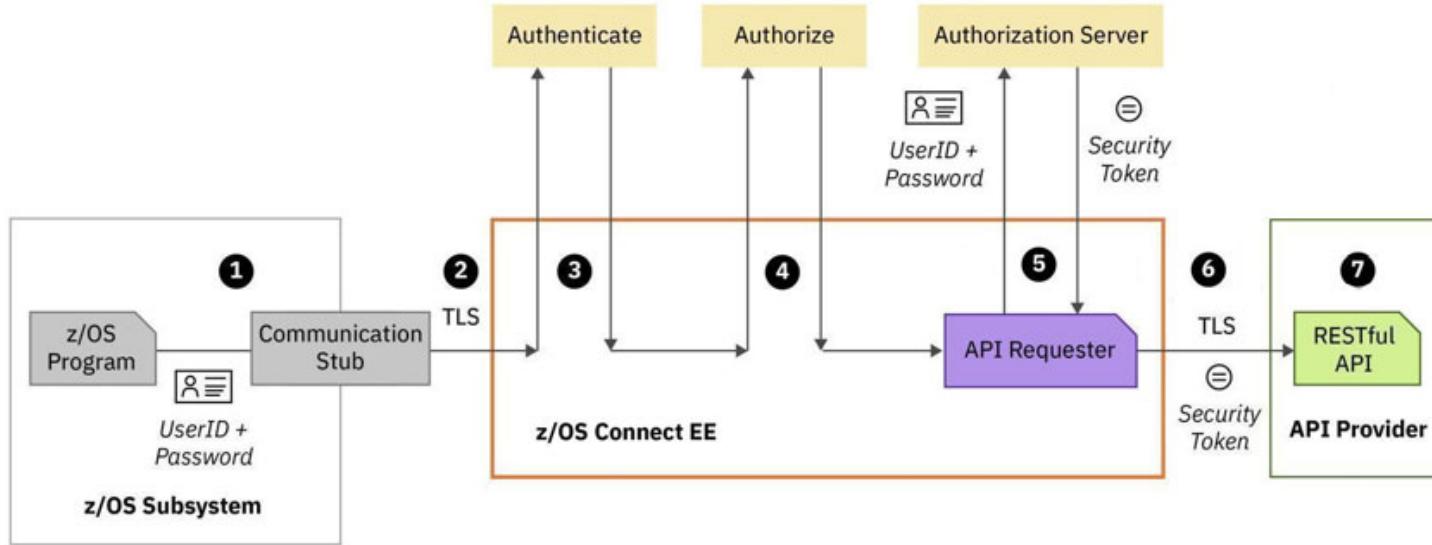
```
<zosconnect_services>
  <service name="mqPut">
    <property name="destination" value="jms/default"/>
    <property name="useCallerPrincipal" value="true"/>
  </service>
</zosconnect_services>
```

Define identity propagation to MQ

**Now let's explore the security options for
outbound API Requester connections
and accessing remote resources**



z/OS Connect EE API Requester security flow



1. A user ID and password can be used for basic authentication by the z/OS Connect EE server
2. Connection between the CICS, IMS, or z/OS application and the z/OS Connect EE server can use TLS
3. Authenticate the CICS, IMS, or z/OS application.
4. Authorize the authenticated user ID to connect to z/OS Connect EE and to perform specific actions on z/OS Connect EE API requesters
5. Pass the user ID and password credentials to an authorization server to obtain a security token.
6. Secure the connection to the external API provider, and provide security credentials such as a security token to be used to invoke the RESTful API
7. The RESTful API runs in the external API provider



Server XML – API Requester - Accessing an API Provider (OpenAPI 2)

```
cscvinc.properties - Notepad
File Edit Format View Help
apiDescriptionFile=./cscvinc.json
dataStructuresLocation=./syslib
apiInfoFileLocation=./syslib
logFileDirectory=./logs
language=COBOL
connectionRef=cscvincAPI
requesterPrefix=csc
```

Server Config

apiRequesterHTTPS.xml

Design Source

```
<server description="API Requester">
  <!-- Enable features -->
  <featureManager>
    <feature>zosconnect:apiRequester-1.0</feature>
  </featureManager>
  <zosconnect_apiRequesters location="/global/zosconnect/resources/apiRequesters">
    <idAssertion>ASSERT_ONLY</idAssertion>
    <apiRequester name="cscvinc_1.0.0" requireSecure="false"/>
  </zosconnect_apiRequesters>
  <zosconnect_endpointConnection id="mqapi">
    host="http://dvipa.washington.ibm.com"
    port="9443"
    authenticationConfigRef="mySAFAuth"
    connectionTimeout="10s"
    receiveTimeout="40s" />
  <zosconnect_endpointConnection id="cscvincAPI">
    host="https://dvipa.washington.ibm.com"
    port="9443"
    connectionTimeout="10s"
    receiveTimeout="40s" />
  <zosconnect_endpointConnection id="miniloancicsAPI">
    host="https://dvipa.washington.ibm.com"
    port="9443"
    authenticationConfigRef="mySAFAuth"
    connectionTimeout="10s"
    receiveTimeout="40s" />
  <zosconnect_authData id="mySAFAuth">
    user="USER1"
    password="user1" />
</server>
```

Server Config

server.xml

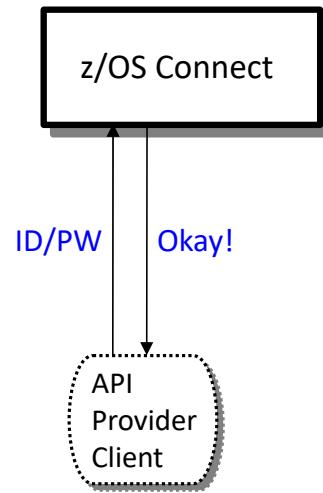
Design Source

```
<!-- To access this server from a remote client add a host attribute to the following
element, e.g. host="*" -->
<httpEndpoint host="*"
  httpPort="9080"
  httpsPort="9443"
  id="defaultHttpEndpoint"/>
```

API Requester – Security from the application to the z/OS Connect server

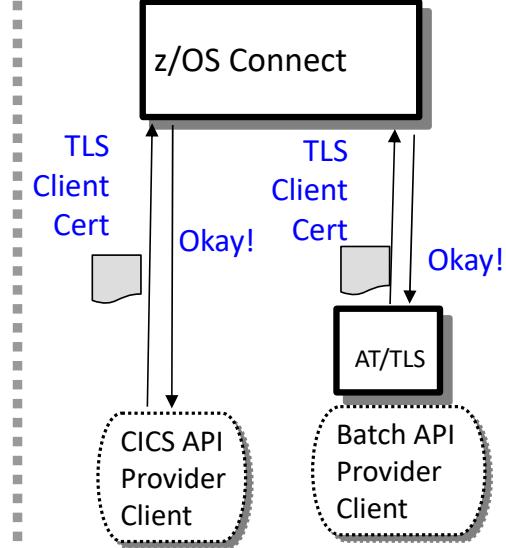
Two options for providing credentials for authentication

Basic Authentication



**Application provides
ID/PW or ID/PassTicket**

Client Certificate

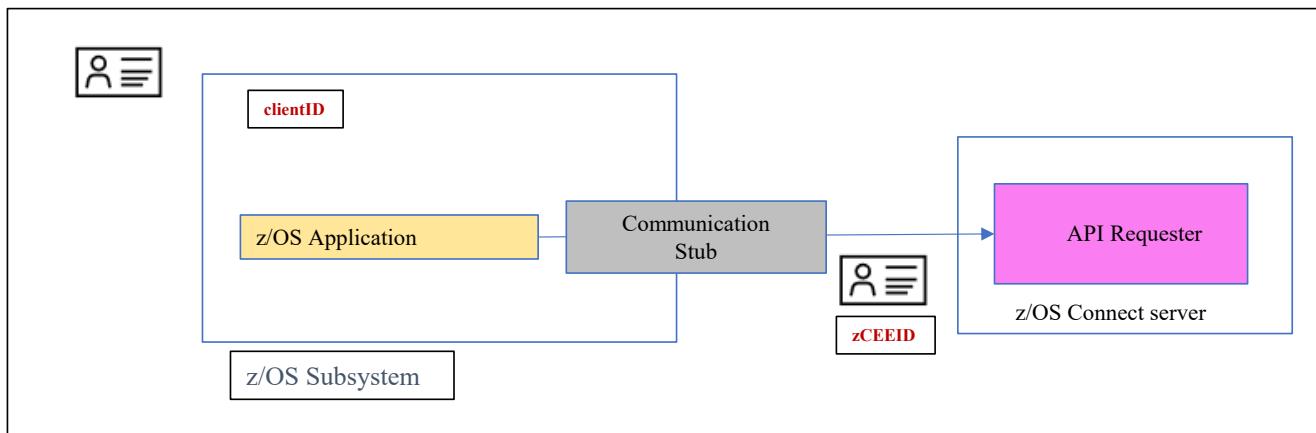


**z/OS Connect requests a
client certificate**

**CICS or AT/TLS supplies a
client certificate**



API Requester - basic authentication and identity assertion



zCEEID – The identity that is used for authenticating connectivity the z/OS subsystem to the zCEE server. It is configured using basic authentication or for CICS, TLS client authentication. For MVS batch, IMS and Db2 stored procedures, the ***zCEEID*** is provided by the environment variable **BAQUSERNAME**. For CICS, the value for ***zCEEID*** is usually provided by the identity mapped to the CICS client certificate.

clientID – the identity under which the z/OS application is executing.

- For CICS, the CICS task identity
- For IMS, the transaction owner
- For batch, the job card USERID

requireAuth	idAssertion	Actions performed by z/OS Connect
true	OFF	Identity assertion is disabled. The zCEE server authenticates <i>zCEEID</i> and checks whether <i>zCEEID</i> has the authority to invoke an API requester.
	ASSERT_SURROGATE	Identity assertion is enabled. The zCEE server authenticates <i>zCEEID</i> and checks whether <i>zCEEID</i> is a surrogate of <i>clientID</i> . If <i>zCEEID</i> is a surrogate of <i>clientID</i> , the server further checks whether <i>clientID</i> has the authority to invoke an API requester; otherwise, a BAQR7114E message occurs.
	ASSERT_ONLY	Identity assertion is enabled. The zCEE server authenticates <i>zCEEID</i> and directly checks whether <i>clientID</i> has the authority to invoke an API requester
false	OFF	Identity assertion is disabled. A BAQR0407W message occurs.
	ASSERT_SURROGATE	Identity assertion is enabled. The zCEE server checks whether <i>clientID</i> has the authority to invoke an API requester, and a warning message occurs to indicate that the ASSERT_ONLY value is used instead of the ASSERT_SURROGATE value.
	ASSERT_ONLY	Identity assertion is enabled. The zCEE server checks whether <i>clientID</i> has the authority to invoke an API requester

```

<zosconnect_zosConnectManager
    requireAuth="true|false"
    requireSecure="true|false"/>

<zosconnect_apiRequesters idAssertion="OFF">

<zosconnect_apiRequester name="cscvinc_1.0.0"
    requireAuth="true|false"
    requireSecure="true|false"/>
    idAssertion="ASSERT_ONLY"> *

<zosconnect_apiRequester name="db2employee_1.0.0"
    requireAuth="true|false"
    requireSecure="true|false"/>
    idAssertion="ASSERT_SURROGATE"> *

</zosconnect_apiRequesters>

```

* Added in V3.0.45



Identity assertion requires setting a program control extended attribute

As root or superuser, set the *libifaedjreg64.so* program control extended attribute bit

- *Permit the server's identity to the required FACILITY resource*

```
PERMIT BPX.SERVER CLASS(FACILITY) ID(LIBSERV) ACCESS(READ)  
SETROPTS RACLIST(FACILITY) REFRESH
```

- *Define a SURROGAT profile for the asserted identity and permit access to connection identity*

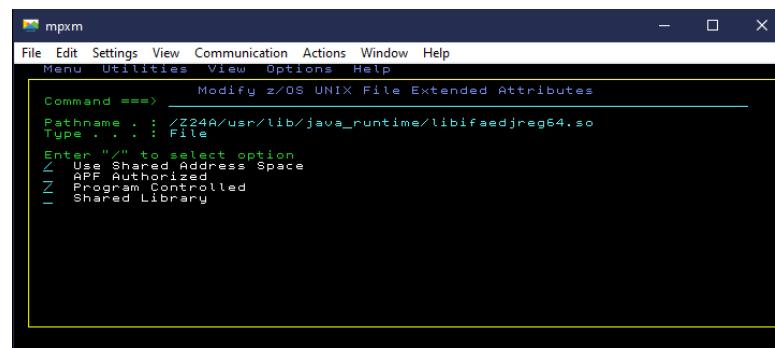
```
RDEFINE SURROGAT clientID.BAQASSRT UACC(NONE) OWNER(SYS1)  
PERMIT clientID.BAQASSRT CLASS(SURROGAT) ACCESS(READ) ID(zCEEID)
```

OR

```
RDEFINE SURROGAT *.BAQASSRT UACC(NONE) OWNER(SYS1)  
PERMIT *.BAQASSRT CLASS(SURROGAT) ACCESS(READ) ID(zCEEID)  
SETROPTS RACLIST(SURROGAT) REFRESH
```

- *Enable the program control bit for Java shared object ifaedjreg64*

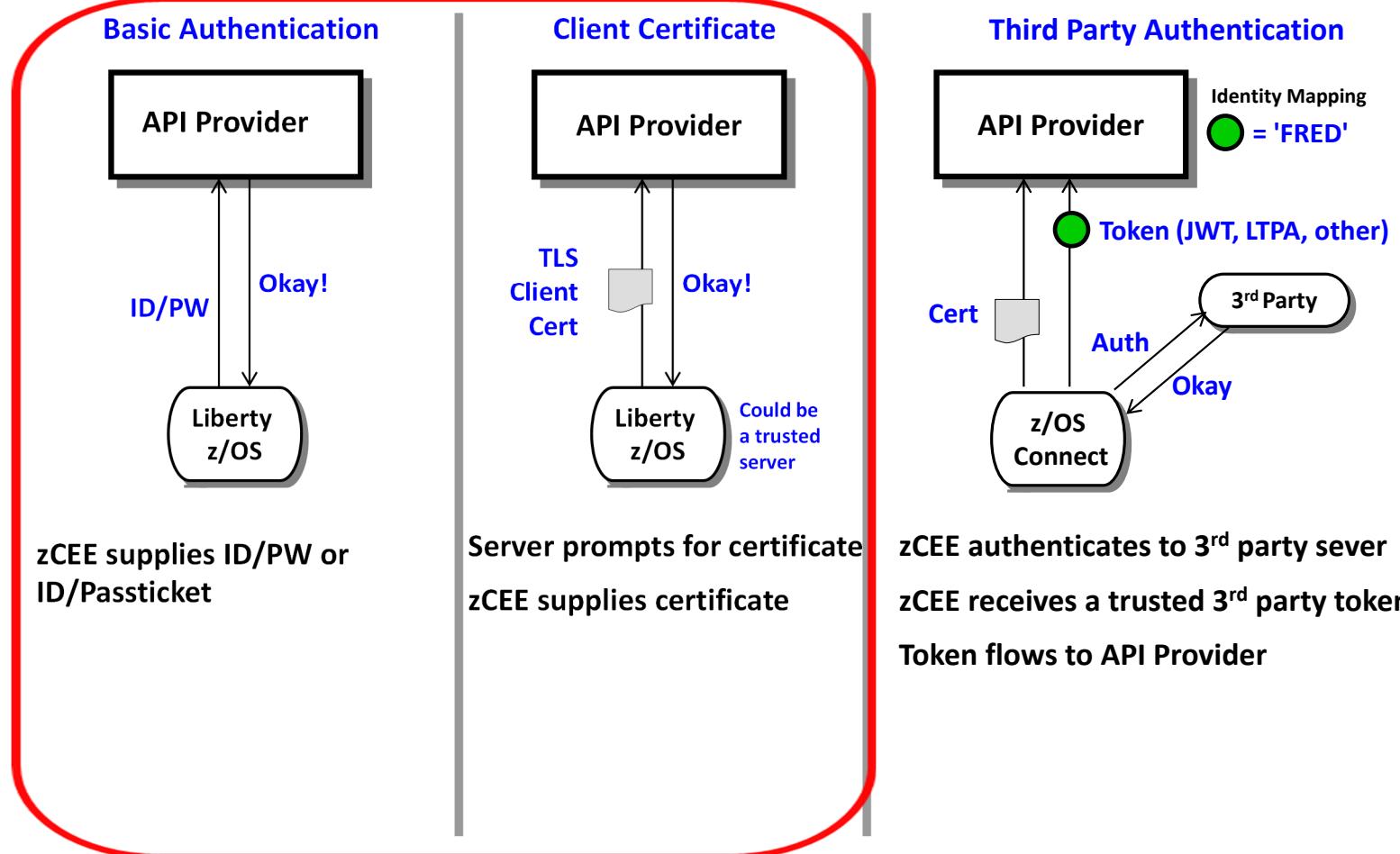
```
su  
cd /usr/lib/java_runtime  
extattr +p libifaedjreg64.so
```





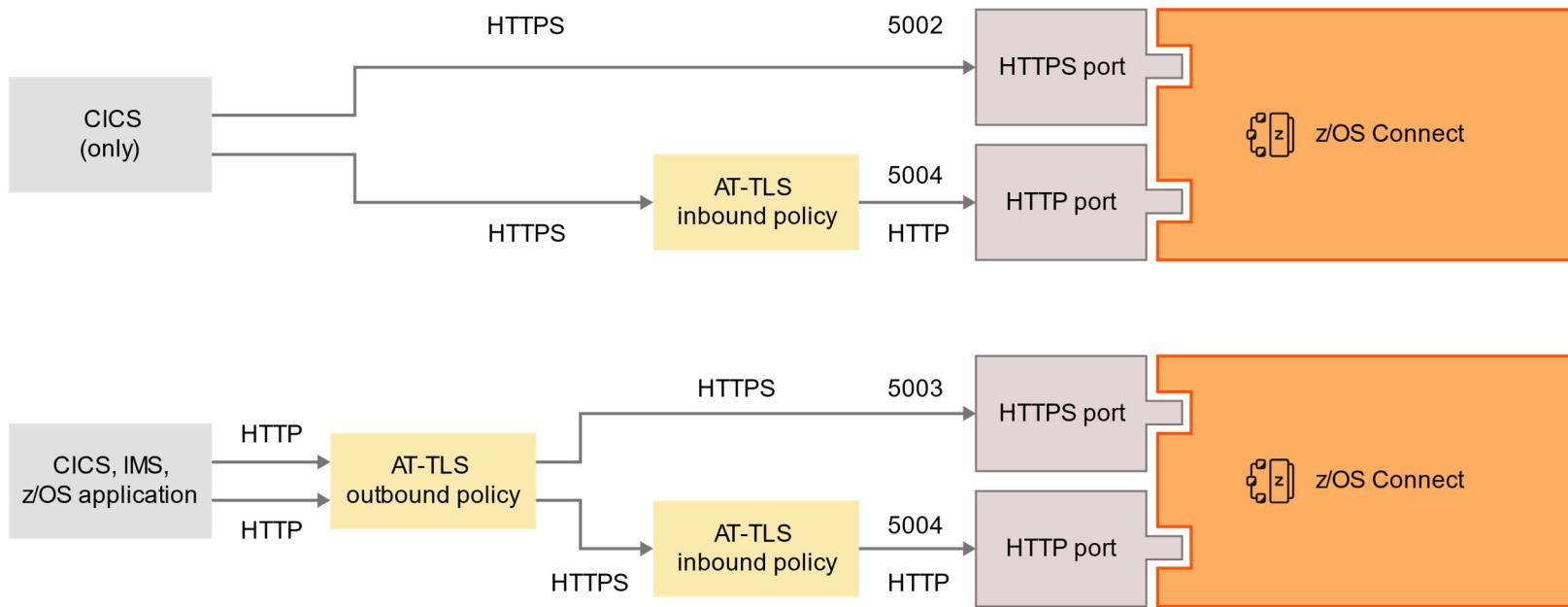
API Requester - API Provider Authentication

Several different ways this can be accomplished:





TLS Connection options from an application to the z/OS Connect server





Tech/Tip: API Requester - HTTP v HTTPS

MVS Batch and IMS with and without an outbound AT-TLS policy

```
CEE0PTS DD *
  POSIX(ON),
  ENVAR("BAQURI=wg31.washington.ibm.com",
  "BAQPORT=9080")
```

```
CEE0PTS DD *
  POSIX(ON),
  ENVAR("BAQURI=wg31.washington.ibm.com",
  "BAQPORT=9443")
```

CICS URIMAPS

The image shows two side-by-side CICS URIMAPS panels. Both panels have a title bar 'WG31' and a menu bar 'File Edit Settings View Communication Actions Window Help'. The left panel is labeled 'CICS RELEASE' and the right panel is labeled 'CICS RELEASE = 0710'. Both panels show an 'OVERTYPE TO MODIFY' screen for a 'CEDA Alter UriMap(BAQURIMP)' command. The configuration parameters are identical in both panels, except for the port number which is circled in red:

Parameter	Value (Left Panel)	Value (Right Panel)	
Urimap	: BAQURIMP	: BAQURIMP	
Group	: SYSGRP	: SYSGRP	
DESCription	==> URIMAP for z/OS Connect EE server	==> URIMAP for z/OS Connect EE server	
Status	==> Enabled	==> Enabled	
USAge	==> Client	==> Client	
	Server Client Pipeline	Server Client Pipeline Atom	
	Jvmserver		
UNIVERSAL RESOURCE IDENTIFIER			
SCHEME	==> HTTP	==> HTTPS	
Port	==> 09120	==> 09443	
HOST	==> wg31.washington.ibm.com	==> wg31.washington.ibm.com	
Path	==> /	==> /	
(Mixed Case)	==>	==>	
	==>	==>	
	==>	==>	
	==>	==>	
+ OUTBOUND CONNECTION POOLING			
	SYSID=CICS APPL	SYSID=CICS APPLID=CICS53Z	
PF 1 HELP 2 COM 3 END	6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11	PF 1 HELP 2 COM 3 END	6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
MB	C	MB	C
Connected to remote server/host wg31 using lu/pool TCP00133 and port 23		Connected to remote server/host wg31 using lu/pool TCP00133 and port 23	

Field BAQ-ZCON-SERVER-URI was added to BAQRINFO in V3.0.37.

MOVE "URIMAP01" TO BAQ-ZCON-SERVER-URI.



Configuring Basic and/or TSL support – z/OS Connect API Requester

Basic authentication with HTTP protocol

```
<zosconnect_endpointConnection id="cscvincAPI"  
    host="http://wg31.washington.ibm.com" port="9080"  
    authenticationConfigRef="myAuthData" />  
  
<zosconnect_authData id="myAuthData"  
    user="zCEEClient" password="secret"/>
```

TLS with HTTPS protocol

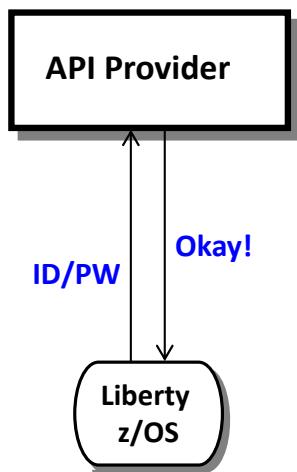
```
<zosconnect_endpointConnection id="cscvincAPI"  
    host="https://wg31.washington.ibm.com" port="9443"  
    authenticationConfigRef="myAuthData" 1  
    sslCertsRef="OutboundSSLSettings" />  
  
<zosconnect_authData id="myAuthData" 1  
    user="zCEEClient" password="secret"/>
```

¹ Optional, if mutual authentication is enabled by the server endpoint

API Requester – Security from the z/OS Connect server to the API provider

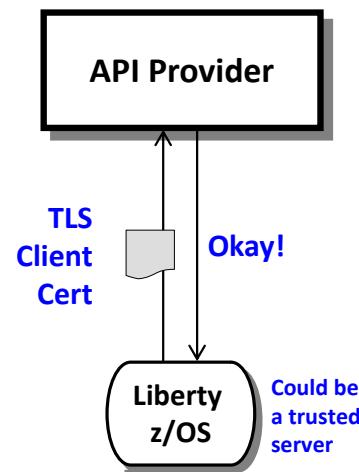
Several different ways this can be accomplished:

Basic Authentication



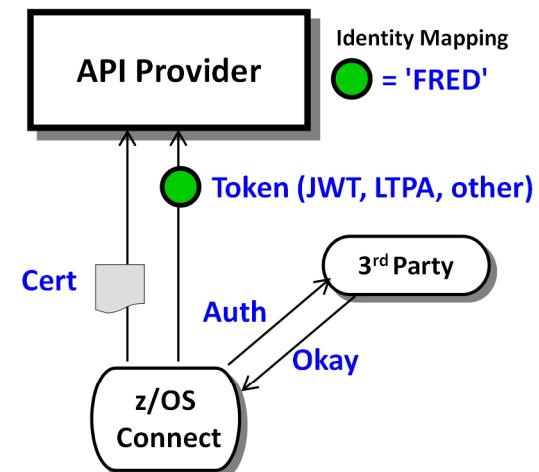
zCEE supplies ID/PW or
ID/Passticket

Client Certificate



Server prompts for certificate
zCEE supplies certificate

Third Party Authentication



zCEE authenticates to 3rd party sever
zCEE receives a trusted 3rd party token
Token flows to API Provider

z/OS Connect API Requester - Token Support

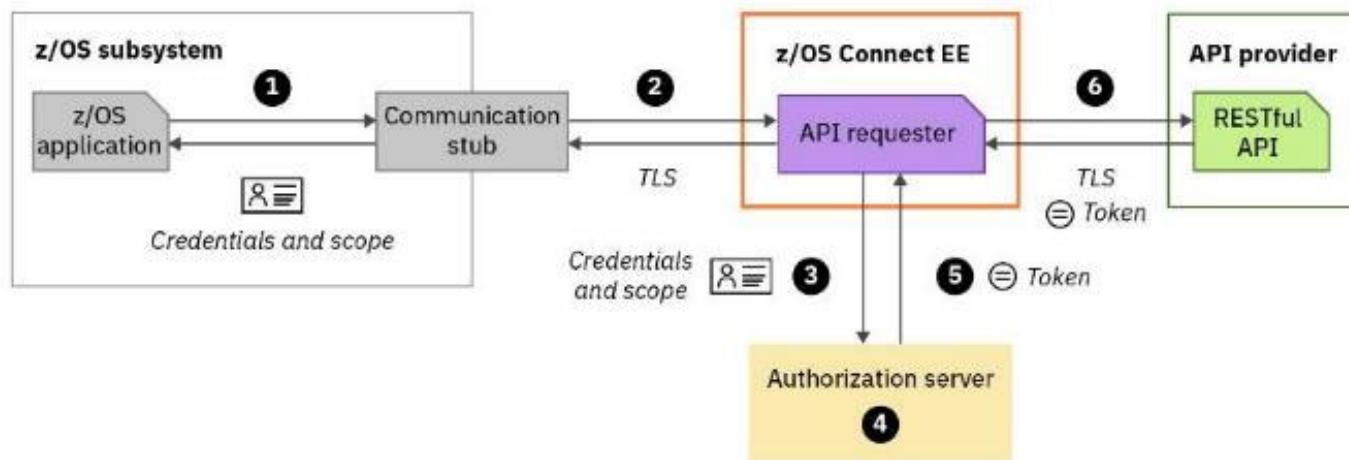


z/OS Connect EE provides *three* ways of calling an API secured with a token

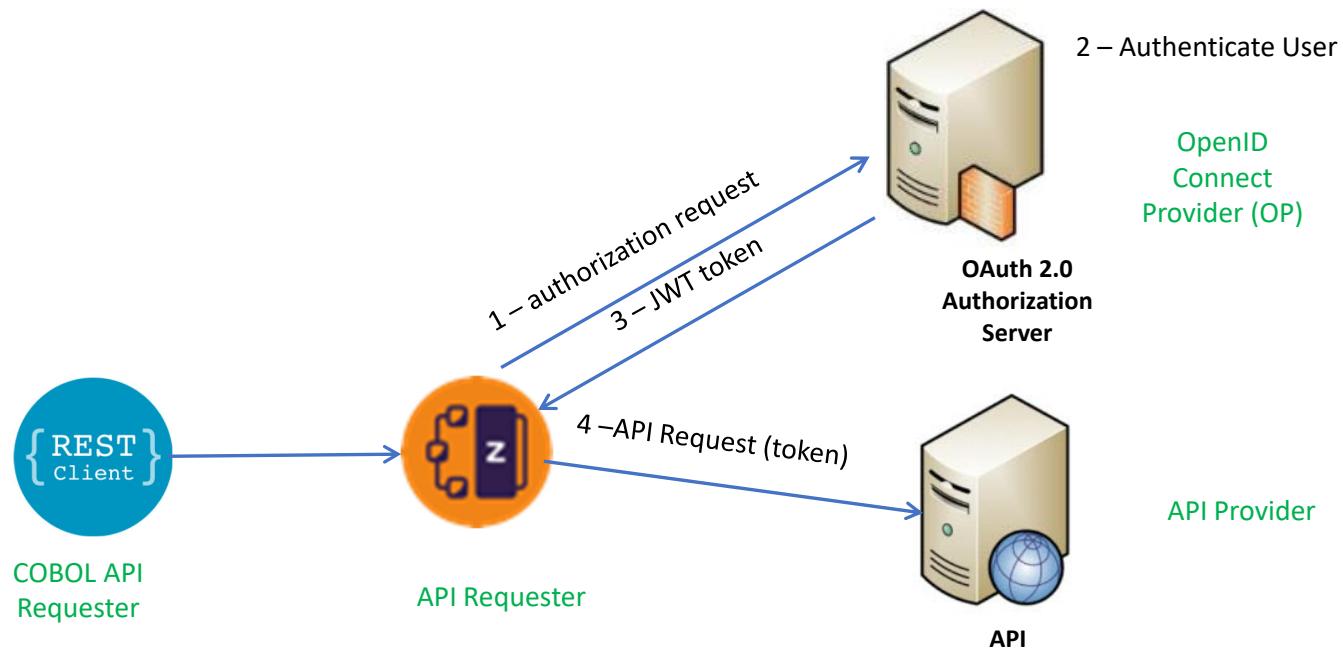
1. Use the OAuth 2.0 support when the request is part of an OAuth 2.0 flow. With OAUTH configured, the token can be an opaque token or a JWT token.
1. In a non-OAuth 2.0 scenario, a JWT token is used in a custom flow, for example: when you need to specify the HTTP verb that is used in the request to the authentication server.
 - When you need to specify the HTTP verb that is used in the request to the authentication server
 - When you need to specify how the JWT is returned from the authentication server (for example, in an HTTP header or in a custom field in a JSON response message).
 - When you need to use a custom header name for sending the JWT to the request endpoint.
3. Use the locally generated JWT support when you need to send a JWT that is generated by the z/OS Connect EE server.



Calling an API with OAuth 2.0 support



z/OS Connect OAuth Flow for API requester



Grant Types:

- client_credentials
- password



OAuth Grant Types Supported by z/OS Connect

client_credentials - the identity associated with the combination of the CICS, IMS, or z/OS application, and the z/OS Connect EE server that calls the RESTful API on behalf of the CICS, IMS, or z/OS application When this grant type is used, the z/OS Connect EE server sends the client credentials and the access scope to the authorization server.

```
<zosconnect_oAuthConfig id="myoAuthConfig"  
    grantType="client_credentials"  
    authServerRef="myoAuthServer"/>
```

password - The identity of the user of the CICS, IMS, or z/OS application, or it might be another entity. When this grant type is used, the z/OS Connect EE server sends the resource owner's credentials, the client credentials, and the access scope to the authorization server.

```
<zosconnect_oAuthConfig id="myoAuthConfig"  
    grantType="password"  
    authServerRef="myoAuthServer"/>
```



Configuring OAuth support – BAQRINFO copy book

```

wg31 master
File Edit Settings View Communication Actions Window Help
Menu Utilities Compilers Help
BROWSE ZCEE30.SBAQC0B(BAQRINFO) Line 0000000028 Col 001 080
Command ==> - Scroll ==> PAGE
01 BAQ-REQUEST-INFO.
03 BAQ-REQUEST-INFO-COMP-LEVEL PIC S9(9) COMP-5 SYNC VALUE 4.
03 BAQ-REQUEST-TINFO-USER
05 BAQ-OAUTH.
07 BAQ-OAUTH-USERNAME PIC X(256).
07 BAQ-OAUTH-USERNAME-LEN PIC S9(9) COMP-5 SYNC
    VALUE 0.
07 BAQ-OAUTH-PASSWORD PIC X(256).
07 BAQ-OAUTH-PASSWORD-LEN PIC S9(9) COMP-5 SYNC
    VALUE 0
07 BAQ-OAUTH-CLIENTID PIC X(256).
07 BAQ-OAUTH-CLIENTID-LEN PIC S9(9) COMP-5 SYNC
    VALUE 0.
07 BAQ-OAUTH-CLIENT-SECRET PIC X(256).
07 BAQ-OAUTH-CLIENT-SECRET-LEN PIC S9(9) COMP-5 SYNC
    VALUE 0
07 BAQ-OAUTH-SCOPE-PTR USAGE POINTER.
07 BAQ-OAUTH-SCOPE-LEN PIC S9(9) COMP-5 SYNC
    VALUE 0.
05 BAQ-AUTHTOKEN.
07 BAQ-TOKEN-USERNAME PIC X(256).
07 BAQ-TOKEN-USERNAME-LEN PIC S9(9) COMP-5 SYNC
    VALUE 0.
07 BAQ-TOKEN-PASSWORD PIC X(256).
07 BAQ-TOKEN-PASSWORD-LEN PIC S9(9) COMP-5 SYNC
    VALUE 0.
05 BAQ-ZCON-SERVER-URI PIC X(256)
    VALUE SPACES.
04/015
Connected to remote server/host wg31z using lu/pool TCP00145

```

Grant Type: *password* - The identity of the user provided by the CICS, IMS, or z/OS application, or it might be another entity. Client_credentials can be supplied by the program or in the server XML configuration.

Grant Type: *client_credentials* - the identity associated with the combination of the CICS, IMS, or z/OS application, and the z/OS Connect EE server that calls the RESTful API on behalf of the CICS, IMS, or z/OS application

Scope is always required.

OAuth 2.0 specification entity	password	client_credentials	Where Set
Client ID	required	Required	server.xml or by application
Client Secret	optional	Required	server.xml or by application
Username	required	N/A	by application
Password	required	N/A	by application



Sample program and JCL

COBOL Application

```
MOVE "ATSOAUTHUSERNAME" to envVariableName.  
PERFORM CALL-CEEENV THRU CALL-CEEENV-END  
MOVE VAR(1:valueLength) to BAQ-OAUTH-USERNAME  
MOVE valueLength TO BAQ-OAUTH-USERNAME-LEN  
MOVE "ATSOATHPASSWORD" to envVariableName.  
PERFORM CALL-CEEENV THRU CALL-CEEENV-END  
MOVE VAR(1:valueLength) to BAQ-OAUTH-PASSWORD  
MOVE valueLength to BAQ-OAUTH-PASSWORD-LEN  
MOVE SPACES      to BAQ-OAUTH-CLIENTID.  
MOVE 0          to BAQ-OAUTH-CLIENTID-LEN.  
MOVE SPACES      to BAQ-OAUTH-CLIENT-SECRET.  
MOVE 0          to BAQ-OAUTH-CLIENT-SECRET-LEN.  
MOVE "openid"    to BAQ-OAUTH-SCOPE.  
MOVE 6          to BAQ-OAUTH-SCOPE-LEN.  
SET BAQ-OAUTH-SCOPE-PTR TO ADDRESS OF BAQ-OAUTH-SCOPE.
```

Note that this example is using environment variables to provide OAuth credentials, as documented in the z/OS Connect Advanced Topics Guide.

Execution JCL

```
//GETAPI EXEC PGM=GETAPIPT,PARM='111111'  
//STEPLIB  DD DISP=SHR,DSN=USER1.ZCEE30.LOADLIB  
//          DD DISP=SHR,DSN=ZCEE30.SBAQLIB  
//          DD DISP=SHR,DSN=JOHNSON.ZCEE.SDFHLOAD  
//SYSOUT   DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
//CEEOPTS  DD *  
POSIX(ON),  
ENVAR ("BAQURI=wg31.washington.ibm.com",  
"BAQPORT=9080",  
"BAQUSERNAME=USER1",  
"ATSAPPL=BBGZDFLT",  
"ATSOAUTHUSERNAME=distuser1",  
"ATSOATHPASSWORD=pwd")
```

Tech/Tip: Accessing environment variables from COBOL application

```
*****  
** Get the BAQ-OAUTH-USERNAME environment variable  
*****  
MOVE "ATSOAUTHUSERNAME" TO envVariableName.  
PERFORM CALL-CEEENV THRU CALL-CEEENV-END  
IF valueLength NOT = 0 THEN  
    MOVE VAR(1:valueLength) TO BAQ-OAUTH-USERNAME  
    MOVE valueLength TO BAQ-OAUTH-USERNAME-LEN  
    DISPLAY "BAQ-OAUTH-USERNAME: "  
        BAQ-OAUTH-USERNAME(1:BAQ-OAUTH-USERNAME-LEN)  
ELSE  
    DISPLAY "BAQ-OAUTH-USERNAME: Not found"  
END-IF.  
*****  
** Get the BAQ-OAUTH-PASSWORD environment variable  
*****  
MOVE "ATSOAUTHPASSWORD" TO envVariableName.  
PERFORM CALL-CEEENV THRU CALL-CEEENV-END  
IF valueLength NOT = 0 THEN  
    MOVE VAR(1:valueLength) TO BAQ-OAUTH-PASSWORD  
    MOVE valueLength TO BAQ-OAUTH-PASSWORD-LEN  
    DISPLAY "BAQ-OAUTH-PASSWORD: "  
        BAQ-OAUTH-PASSWORD(1:BAQ-OAUTH-PASSWORD-LEN)  
ELSE  
    DISPLAY "BAQ-OAUTH-PASSWORD: Not found"
```

```
01 functionCode PIC 9(9) BINARY.  
01 envVariableNameLength PIC 9(9) BINARY.  
01 envVariableName PIC X(255).  
01 valueLength PIC 9(9) BINARY.  
01 valuePointer POINTER.  
01 ws-length PIC 9(3).  
  
01 feedbackCode.  
02 CONDITION-TOKEN-VALUE.  
COPY CEEIGZCT.  
    03 CASE-1-CONDITION-ID.  
        04 SEVERITY      PIC S9(4) BINARY.  
        04 MSG-NO       PIC S9(4) BINARY.  
    03 CASE-SEV-CTL   PIC X.  
    03 FACILITY-ID   PIC XXX.  
    02 I-S-INFO        PIC S9(9) BINARY.  
01 VAL          PIC X(255).
```

```
CALL-CEEENV.  
    MOVE 1 TO functionCode.  
    MOVE ZERO TO ws-length.  
    INSPECT FUNCTION REVERSE (envVariableName)  
        TALLYING ws-length FOR LEADING SPACES.  
    COMPUTE envVariableNameLength =  
        LENGTH OF envVariableName - ws-length.  
    MOVE " " TO VAL.  
    MOVE 0 TO valueLength.  
    CALL "CEEENV" USING functionCode,  
        envVariableNameLength,  
        envVariableName,  
        valueLength,  
        valuePointer,  
        feedbackCode.  
  
    IF valueLength NOT = 0 THEN  
        SET ADDRESS OF VAR TO valuePointer .  
  
CALL-CEEENV-END.,
```

Configuring OAuth support – z/OS Connect API Requester



```
<zosconnect_endpointConnection id="cscvincAPI"
    host="http://wg31.washington.ibm.com" port="9080"
    authenticationConfigRef="myoAuthConfig"/>

<zosconnect_oAuthConfig id="myoAuthConfig"
    grantType="client_credentials|password"
    authServerRef="myoAuthServer"/>

<zosconnect_authorizationServer id="myoAuthServer"
    tokenEndpoint=https://wg31.washington.ibm.com:59443/oidc/endpoint/OP/token1
    basicAuthRef="tokenCredential" 2
    sslCertsRef="OutboundSSLSettings" />

<zosconnect_authData id="tokenCredential" 2
    user="zCEEClient" password="secret"/>
```

```
openidConnectProvider id="OP"
    signatureAlgorithm="RS256"
    keyStoreRef="jwtStore"
    oauthProviderRef="OIDCssl" >
</openidConnectProvider>
```

¹See URL https://www.ibm.com/support/knowledgecenter/SS7K4U/liberty/com.ibm.websphere.wlp.zseries.doc/ae/twlp_oidc_token_endpoint.html

² These credentials can be specified by the application



Security Scenarios

```
BAQ-OAUTH-USERNAME: distuser1  
BAQ-OAUTH-PASSWORD: pwd  
EmployeeNumber: 111111  
EmployeeName: C. BAKER  
USERID: USER1
```

distuser1 is mapped to RACF identity USER1 who has full access

```
BAQ-OAUTH-USERNAME: distuserx  
BAQ-OAUTH-PASSWORD: pwd  
Error code: 00000500  
Error msg:{ "errorMessage": "BAQR1092E: Authentication or authorization failed for the z/OS Connect EE server." }
```

distuserx is unknown by the OAuth Provider

```
BAQ-OAUTH-USERNAME: auser  
BAQ-OAUTH-PASSWORD: pwd  
Error code: 0000000403  
rror msg:{ "errorMessage": "BAQR1144E: Authentication or authorization failed for the z/OS Connect EE server." }  
Syslog:  
ICH408I USER(ATSSERV ) GROUP(ATSGRP ) NAME(LIBERTY SERVER  
DISTRIBUTED IDENTITY IS NOT DEFINED:  
auser zCEERealm
```

auser is not mapped to a valid RACF identity

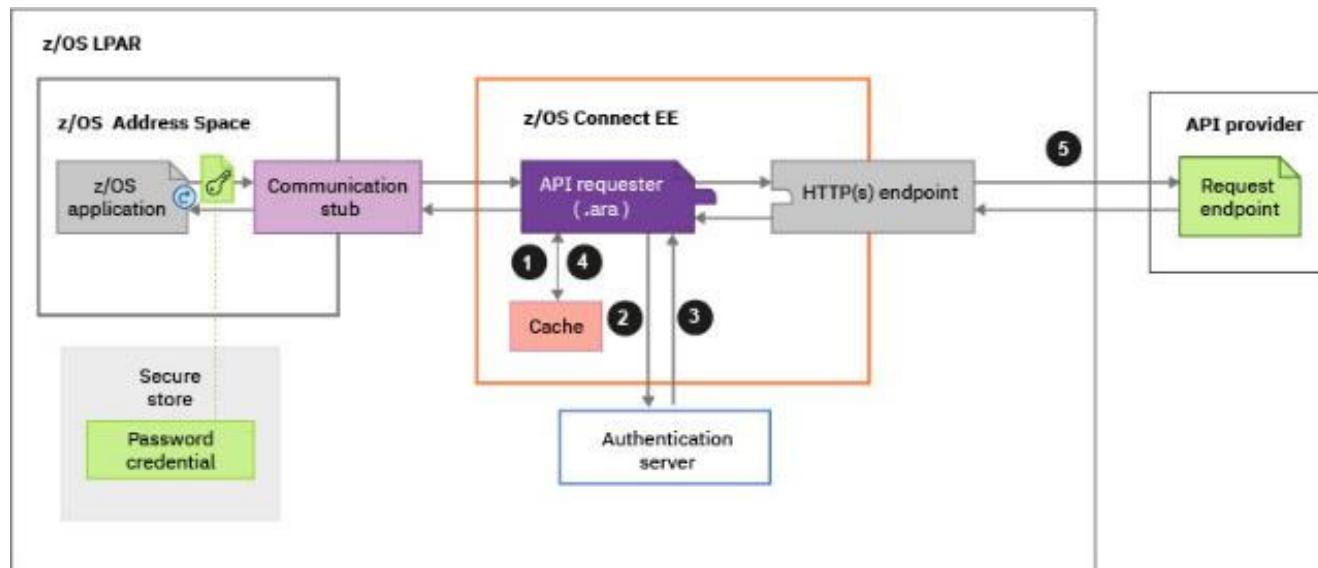
```
BAQ-OAUTH-USERNAME: distuser2  
BAQ-OAUTH-PASSWORD: pwd  
Error code: 0000000403  
Error msg:{ "errorMessage": "BAQR1144E: Authentication or authorization failed for the z/OS Connect EE server." }  
Syslog:  
ICH408I USER(USER2 ) GROUP(SYS1 ) NAME(WORKSHOP USER2  
ATSZDFLT.zos.connect.access.roles.zosConnectAccess  
CL(EJBROLE )  
INSUFFICIENT ACCESS AUTHORITY  
ACCESS INTENT(READ ) ACCESS ALLOWED(NONE )
```

distuser2 is mapped to RACF identity USER2 which has no access to the EJBRole protecting z/OS Connect

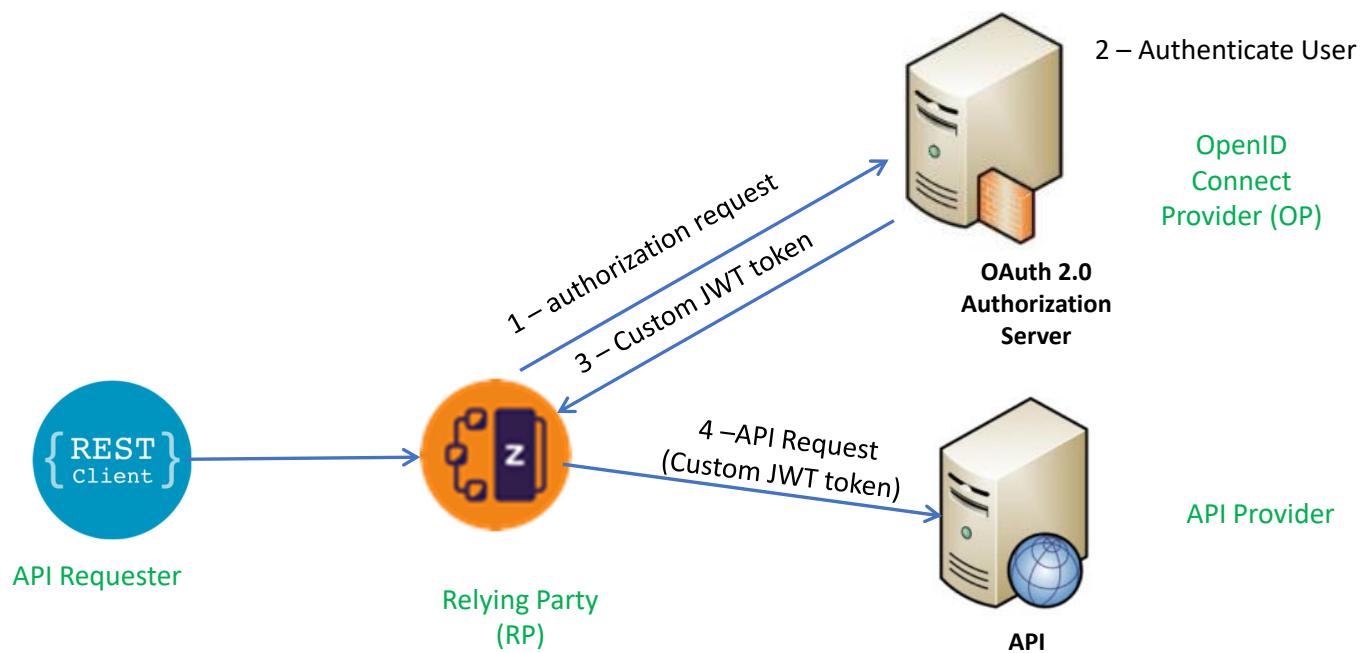


Calling an API with using a JWT custom flow

- ❑ In a non-OAuth 2.0 scenario, a JWT token is used in a custom flow, for example:
 - When you need to specify the HTTP verb that is used in the request to the authentication server.
 - When you need to specify how the JWT is returned from the authentication server (for example, in an HTTP header or in a custom field in a JSON response message).
 - When you need to use a custom header name for sending the JWT to the request endpoint.



z/OS Connect OAuth Custom Flow





API Requester – JWT Custom flow

```
BROWSE ZCEE30.SBAQCOB(BAQRINFO) Line 0000000028 Col 001 080
Command ==> - Scroll ==> PAGE
01 BAQ-REQUEST-INFO.
  03 BAQ-REQUEST-INFO-COMP-LEVEL PIC S9(9) COMP-5 SYNC VALUE 4.
  03 BAQ-REQUEST-INFO-USER.
    05 BAQ-OAUTH.
      07 BAQ-OAUTH-USERNAME          PIC X(256).
      07 BAQ-OAUTH-USERNAME-LEN      PIC S9(9) COMP-5 SYNC
                                      VALUE 0.
      07 BAQ-OAUTH-PASSWORD          PIC X(256).
      07 BAQ-OAUTH-PASSWORD-LEN      PIC S9(9) COMP-5 SYNC
                                      VALUE 0.
      07 BAQ-OAUTH-CLIENTID          PIC X(256).
      07 BAQ-OAUTH-CLIENTID-LEN      PIC S9(9) COMP-5 SYNC
                                      VALUE 0.
      07 BAQ-OAUTH-CLIENT-SECRET    PIC X(256).
      07 BAQ-OAUTH-CLIENT-SECRET-LEN PIC S9(9) COMP-5 SYNC
                                      VALUE 0.
      07 BAQ-OAUTH-SCOPE-PTR        USAGE POINTER.
      07 BAQ-OAUTH-SCOPE-LEN         PIC S9(9) COMP-5 SYNC
                                      VALUE 0.

  05 BAQ-AUTHTOKEN.
    07 BAQ-TOKEN-USERNAME          PIC X(256).
    07 BAQ-TOKEN-USERNAME-LEN      PIC S9(9) COMP-5 SYNC
                                      VALUE 0.
    07 BAQ-TOKEN-PASSWORD          PIC X(256).
    07 BAQ-TOKEN-PASSWORD-LEN      PIC S9(9) COMP-5 SYNC
                                      VALUE 0.
  05 BAQ-ZCON-SERVER-URI          PIC X(256)
                                      VALUE SPACES.

MA A
Connected to remote server/host wg31z using lu/pool TCP00145 04/015
```

BAQRINFO copy book

COBOL application

```
MOVE "ATSTOKENUSERNAME" to envVariableName.
PERFORM CALL-CEEENV THRU CALL-CEEENV-END
MOVE VAR(1:valueLength) to BAQ-TOKEN-USERNAME
MOVE valueLength TO BAQ-TOKEN-USERNAME-LEN
MOVE "ATSTOKENPASSWORD" to envVariableName.
PERFORM CALL-CEEENV THRU CALL-CEEENV-END
MOVE VAR(1:valueLength) to BAQ-TOKEN-PASSWORD
MOVE valueLength to BAQ-TOKEN-PASSWORD-LEN
```

Note that this example is using environment variables to provide token credentials, as documented in the z/OS Connect Advanced Topics Guide.



Configuring JWT Custom flow

```
<zosconnect_endpointConnection id="cscvincAPI"
    host="http://wg31.washington.ibm.com" port="9080"
    authenticationConfigRef="myJWTConfig"/>

<zosconnect_authToken id="myJWTConfig" authServerRef="myJWTServer"
    header="myJWT-header-name"
    <tokenRequest/>      See next slide
    <tokenReponse/>      See next slide
</zosconnect_authToken>

<zosconnect_authorizationServer id="myJWTServer"
    tokenEndpoint=https://wg31.washington.ibm.com:59443/oidc/endpoint/OP/token1
    basicAuthRef="tokenCredential" 2
    sslCertsRef="OutboundSSLSettings" />

<zosconnect_authData id="tokenCredential" 2
    user="zCEEClient" password="secret"/>
```

¹See URL https://www.ibm.com/support/knowledgecenter/SS7K4U_liberty/com.ibm.websphere.wlp.zseries.doc/ae/twlp_oidc_token_endpoint.html

² These credentials can be specified by the application



Configuring Custom JWT flow

Request Token Example 1

```
<tokenRequest  
    credentialLocation="header"  
    header="Authorization"  
    requestMethod="GET" />
```

Response Token

```
<tokenResponse  
    tokenLocation="header"  
    header="JWTAuthorization" />
```

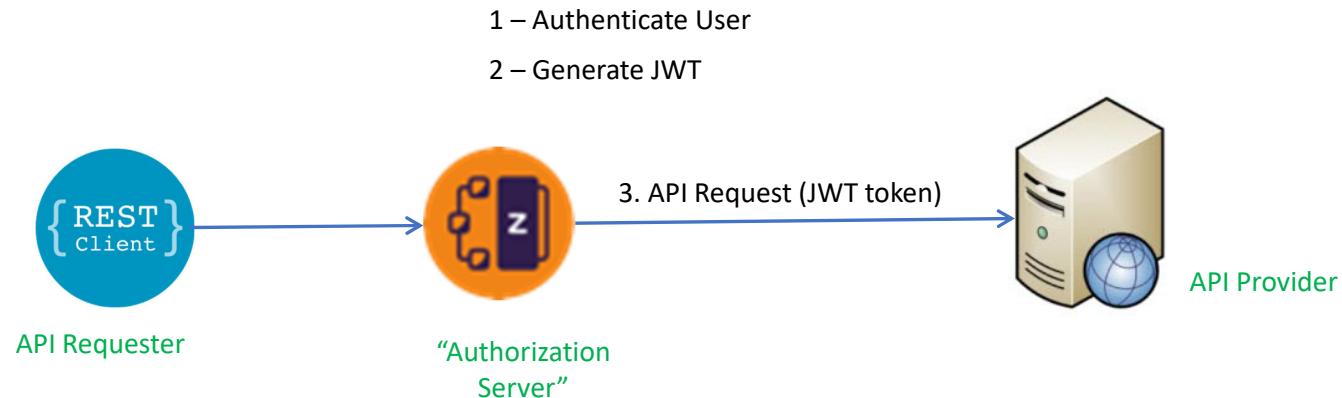
Response Token Example 2

```
<tokenRequest credentialLocation="body"  
    requestMethod="POST"  
    // Use XML escaped characters in requestBody  
    requestBody="
```

Response Token

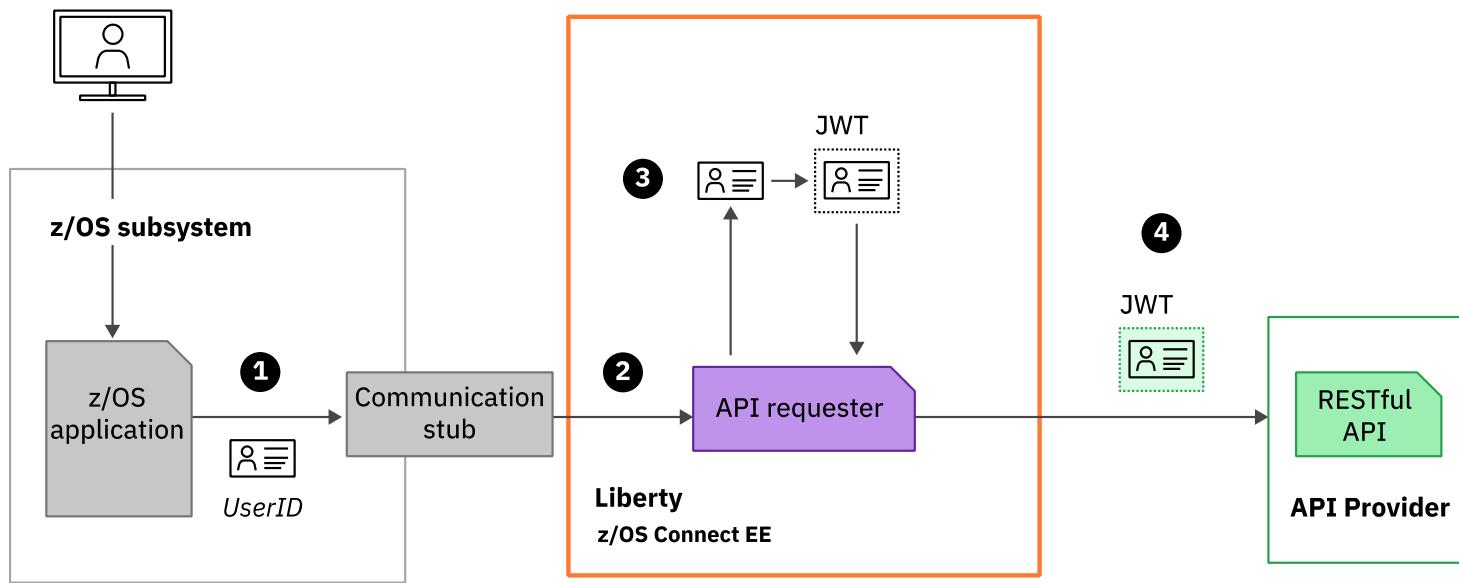
```
<tokenResponse  
    tokenLocation="body"  
    responseFormat="JSON"  
    tokenPath="$&.tokenname" />
```

z/OS Connect JWT Generation – V3.0.43





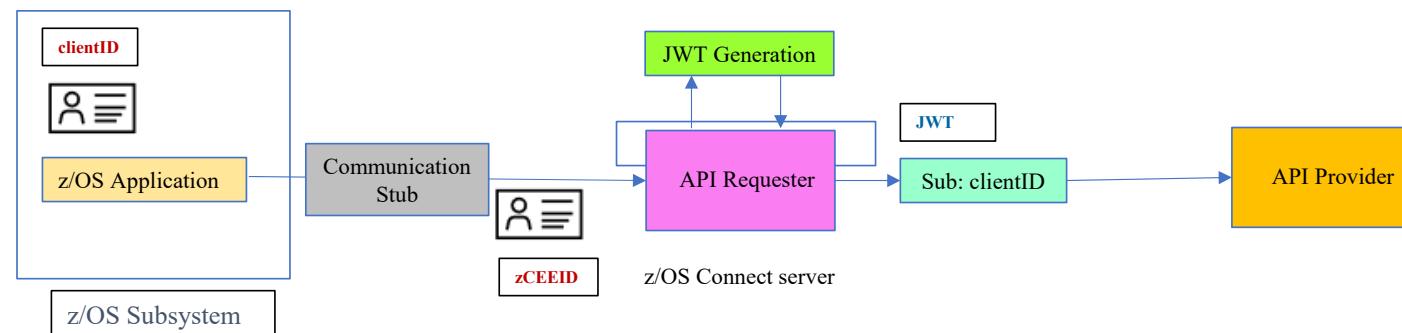
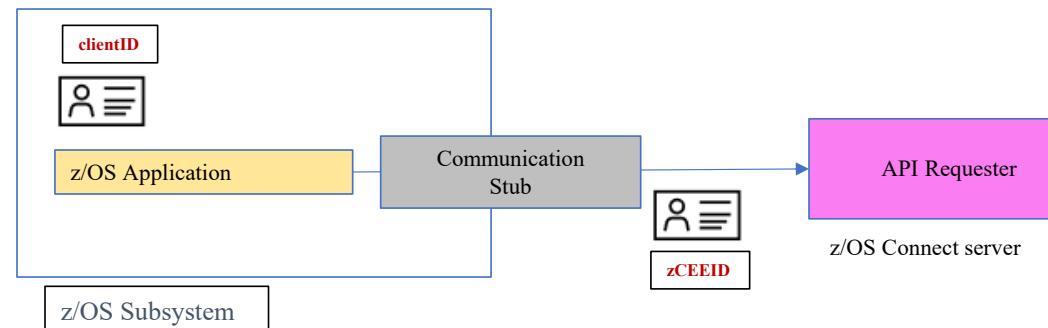
JWT Generation



- 1** Communication stub extracts the ID from the application environment
- 2** z/OS Connect generates a JWT token containing the z/OS application asserted user ID
- 3** The JWT is used to authorise the request to the API endpoint



API Requester - authentication with identity assertion and JWT generation



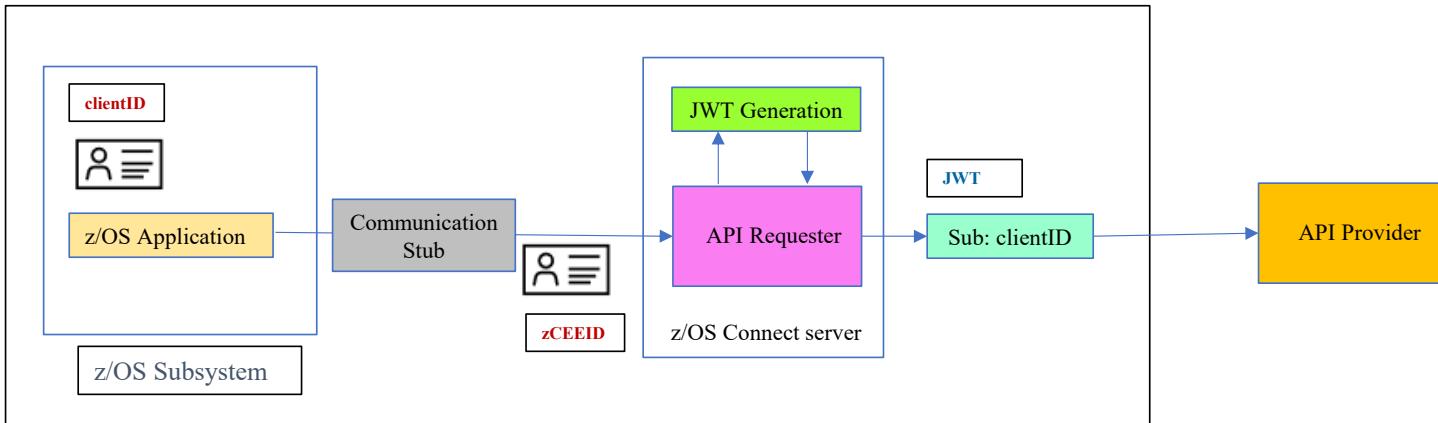
zCEEID – The identity that is used for authenticating connectivity the z/OS subsystem to the zCEE server. It is configured using basic authentication or for CICS, TLS client authentication.

clientID – the identity under which the z/OS application is executing.

- For CICS, the task owner
- For IMS, the transaction owner
- For batch, the job owner



API Requester – JWT Generation



zCEEID – The identity that is used for authenticating connectivity the z/OS subsystem to the zCEE server. It is configured using basic authentication or for CICS, TLS client authentication.

clientID – the identity under which the z/OS application is executing.

- For CICS, the task owner
- For IMS, the transaction owner
- For batch, the job owner

requireAuth	idAssertion	Actions performed by z/OS Connect
true	ASSERT_SURROGATE	Identity assertion is enabled. The zCEE server authenticates <i>zCEEID</i> and checks whether <i>zCEEID</i> is a surrogate of <i>clientID</i> . If <i>zCEEID</i> is a surrogate of <i>clientID</i> , the server further checks whether <i>clientID</i> has the authority to invoke an API requester; otherwise, a BAQR7114E message occurs.
	ASSERT_ONLY	Identity assertion is enabled. The zCEE server authenticates <i>zCEEID</i> and directly checks whether <i>clientID</i> has the authority to invoke an API requester
false	ASSERT_SURROGATE	Identity assertion is enabled. The zCEE server checks whether <i>clientID</i> has the authority to invoke an API requester, and a warning message occurs to indicate that the ASSERT_ONLY value is used instead of the ASSERT_SURROGATE value.
	ASSERT_ONLY	Identity assertion is enabled. The zCEE server checks whether <i>clientID</i> has the authority to invoke an API requester



JWT generation requires setting a program control extended attribute

As root or superuser, set the *libifaedjreg64.so* program control extended attribute bit

- *Permit the server's identity to the required FACILITY resource*

```
PERMIT BPX.SERVER CLASS(FACILITY) ID(LIBSERV) ACCESS(READ)  
SETROPTS RACLIST(FACILITY) REFRESH
```

- *Define a SURROGAT profile for the asserted identity and permit access to connection identity*

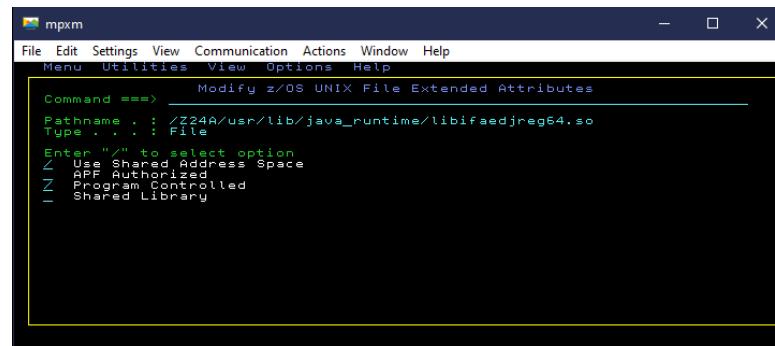
```
RDEFINE SURROGAT clientID.BAQASSRT UACC(NONE) OWNER(SYS1)  
PERMIT clientID.BAQASSRT CLASS(SURROGAT) ACCESS(READ) ID(zCEEID)
```

OR

```
RDEFINE SURROGAT *.BAQASSRT UACC(NONE) OWNER(SYS1)  
PERMIT *.BAQASSRT CLASS(SURROGAT) ACCESS(READ) ID(zCEEID)  
SETROPTS RACLIST(SURROGAT) REFRESH
```

- *Enable the program control bit for Java shared object ifaedjreg64*

```
su  
cd /usr/lib/java_runtime  
extattr +p libifaedjreg64.so
```





Configuring JWT Generation support

```
<zosconnect_endpointConnection id="conn"
    host="http://api.server.com" port="8080"
    authenticationConfigRef="jwtConfig" />

<zosconnect_authTokenLocal id="jwtConfig"
    tokenGeneratorRef="jwtBuilder"
    header="Authorization" >
    <claims>{ "name":"JohnSmith",
        "ID":"1234567890" }
    </claims> One or more Public claim (e.g., aud,exp,nbf,iat,jti) or
one or more private claims

<jwtBuilder id="jwtBuilder"
    scope="scope1"
    audiences="myApp1"
    jti="true"
    signatureAlgorithm="RS256"
    keyStoreRef="myKeyStore"
    keyAlias="jwtsigner"
    issuer="z/OS Connect EE Default"/>
```

The "sub" claim value will be application asserted user ID.



Configuring JWT Generation support

```
<zosconnect_endpointConnection id="conn1"  
    host="http://api.server.com" port="8080"  
    authenticationConfigRef="jwtConfig" />  
<zosconnect_endpointConnection id="conn2"  
    host="http://api.server.com" port="8080"  
    authenticationConfigRef="jwtConfig" />  
<zosconnect_authTokenLocal id="jwtConfig"  
    tokenGeneratorRef="jwtBuilder"  
    header="Authorization" >  
    <claims>{"scope":"Scope1"}</claims>  
<zosconnect_authTokenLocal id="jwtConfig"  
    tokenGeneratorRef="jwtBuilder"  
    header="Authorization" >  
    <claims>{"scope":"Scope2"}</claims>  
<jwtBuilder id="jwtBuilder"  
    scope="scope"  
    audiences="myApp1"  
    jti="true"  
    signatureAlgorithm="RS256"  
    keyStoreRef="myKeyStore"  
    keyAlias="jwtsigner"  
    issuer="z/OS Connect EE Default"/>
```



server XML Configuration

```
→<jwtBuilder id="jwtBuilder"
    scope="scope1"
    audiences="myApp1"
    jti="true"
    signatureAlgorithm="RS256"
    keyStoreRef="myKeyStore"
    keyAlias="jwtSigner"
    issuer="z/OS Connect EE Default"/>

→<zosconnect_authTokenLocal id="jwtConfig"
    tokenGeneratorRef="jwtBuilder"
    header="JWTAuthorization" >
    <claims>{"name":"JohnSmith,
        "ID":"1234567890"}</claims>
</zosconnect_authTokenLocal >
<zosconnect_endpointConnection id="conn"
    host="http://api.server.com" port="8080"
    authenticationConfigRef="jwtConfig" />
```

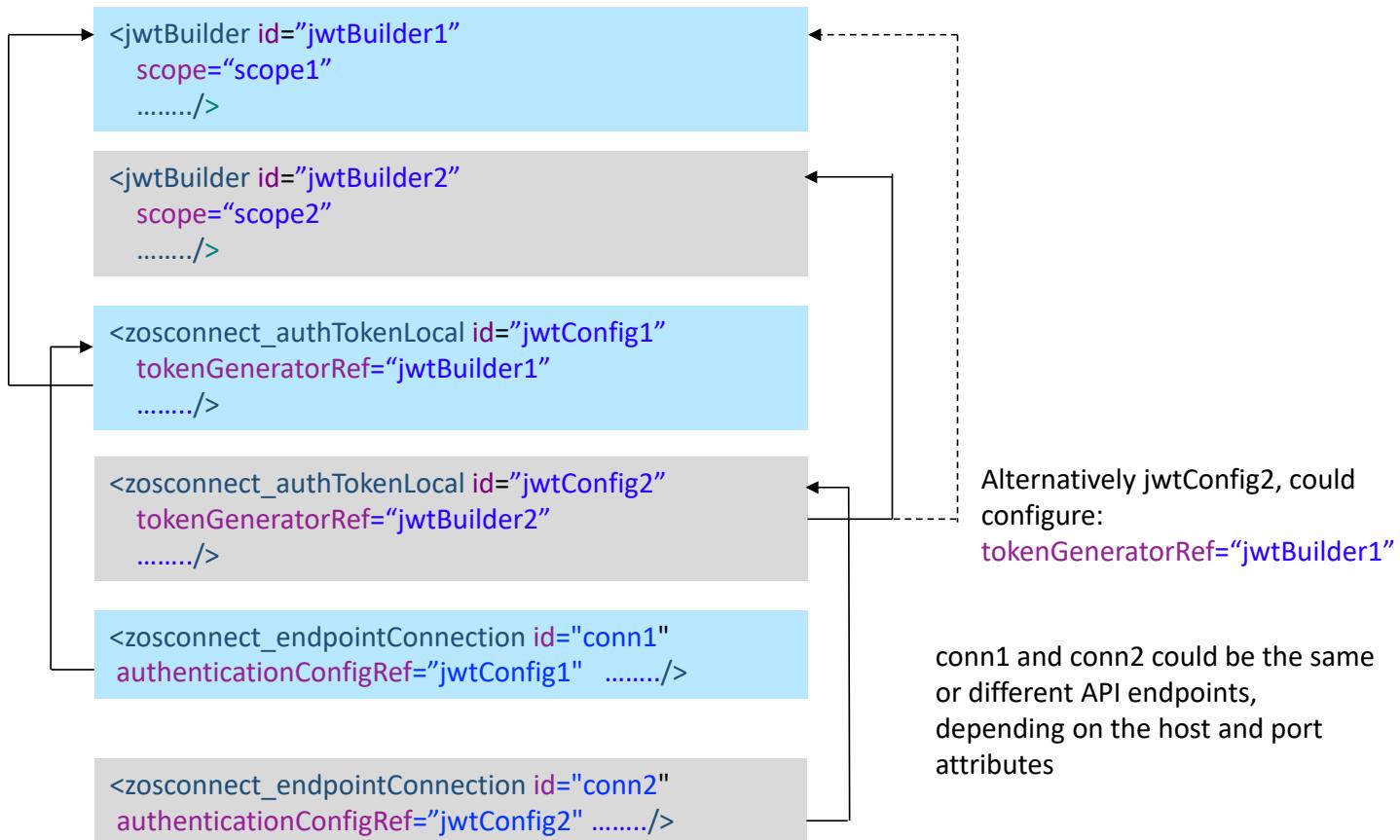
Configure the Liberty jwtBuilder element in server.xml.

Configure the zosconnect_authTokenLocal element, specifying any additional private claims required and the name of the header used to send the JWT to the endpoint.

header default value is Authorization

Finally, reference the JWT configuration from the zosconnect_endpointConnection element.

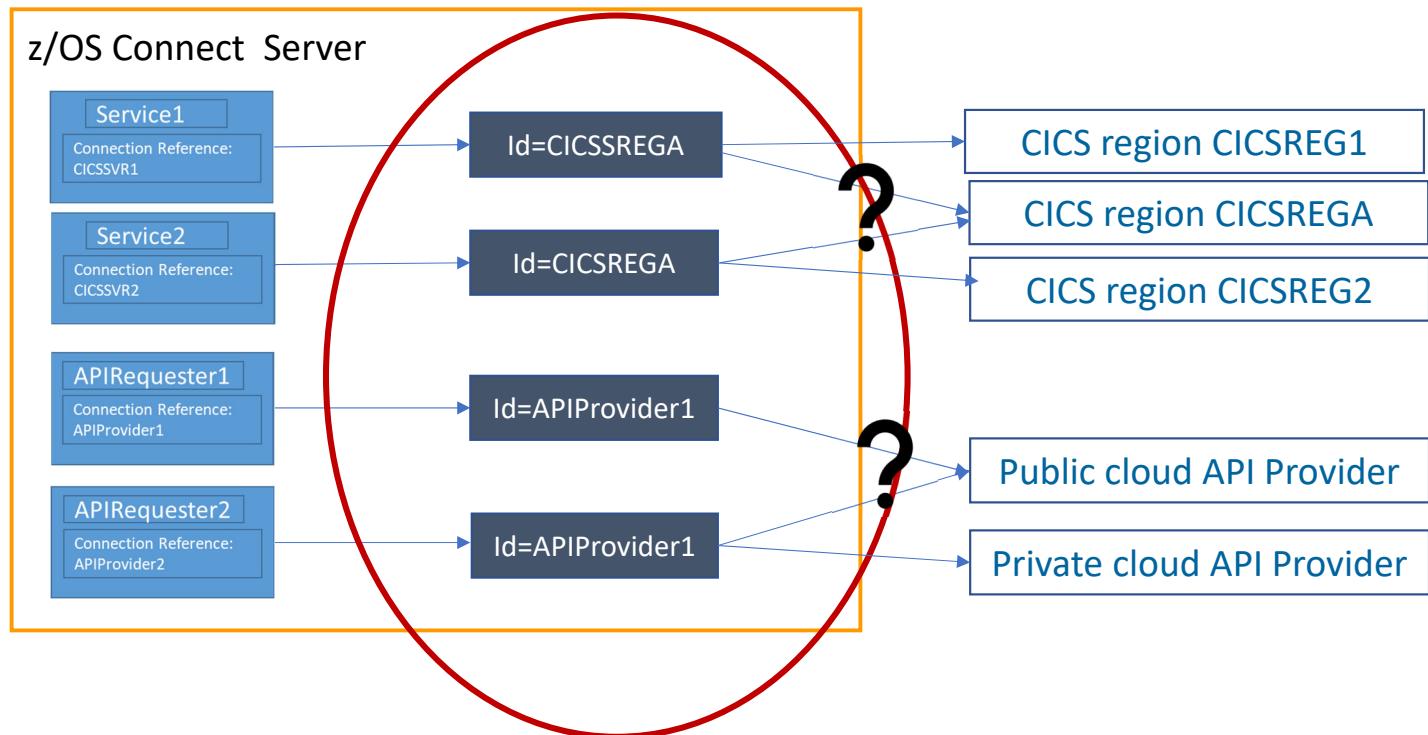
Using different claims for different API endpoints





Use naming conventions for connection references

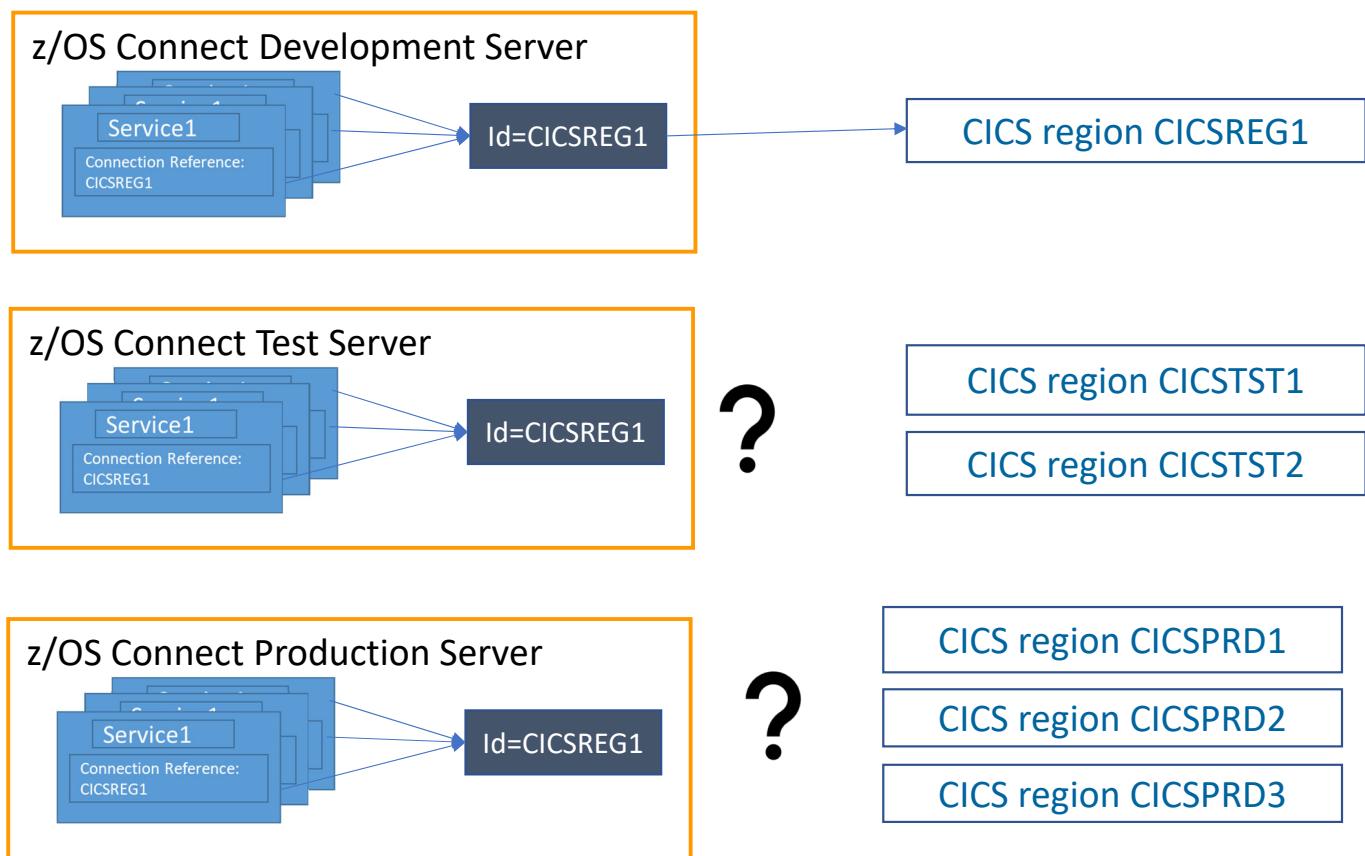
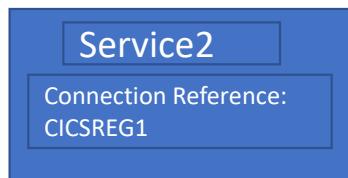
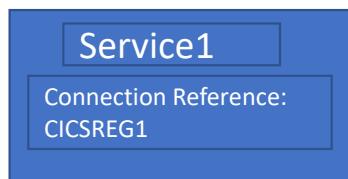
Use application meaningful names or an extendable convention for connection reference names





Use naming conventions for service and endpoint connection references (OpenAPI 2)

Don't couple service and API requester connection names to specific systems or endpoints



Useful Liberty functions/features and MVS commands



Use the adminCenter-1.0 feature to update the server XML from a browser

Administrators can use a web interface to maintain the server XML configuration.

The screenshot shows a web-based configuration tool titled "Server Config". The title bar includes icons for a folder, a user profile, and a save operation. The main area is titled "adminCenter.xml" and has tabs for "Design" and "Source". The "Source" tab is selected, displaying the following XML code:

```
1<server description="Admin Center">
2
3  <!-- Enable features -->
4  <featureManager>
5    <feature>adminCenter-1.0</feature>
6  </featureManager>
7
8  <remoteFileAccess>
9    <writeDir>${server.config.dir}</writeDir>
10 </remoteFileAccess>
11
12</server>
13
```

```
RDEFINE EJBROLE BBGZDFLT.com.ibm.ws.management.security.resource.Administrator OWNER(SYS1) UACC(NONE)
RDEFINE EJBROLE BBGZDFLT.com.ibm.ws.management.security.resource.Reader OWNER(SYS1) UACC(NONE)
```

```
PERMIT BBGZDFLT.com.ibm.ws.management.security.resource.Administrator CLASS(EJBROLE) ID(FRED) ACCESS(READ)
PERMIT BBGZDFLT.com.ibm.ws.management.security.resource.Reader CLASS(EJBROLE) ID(FRED) ACCESS(READ)
```

```
SETR RACLIST(EJBROLE) REFRESH
```



Use Liberty's “adminCenter” Feature to update server XML

- Web browser interface to the server’s configuration files

The screenshot shows the IBM Liberty adminCenter interface for managing server configuration files. The main window title is "Server Config" and the file being edited is "server.xml". The interface has two tabs: "Design" and "Source". The "Source" tab is currently active, showing the XML code for the configuration.

In the XML code, the "zosconnect_apiRequester" element is highlighted with a green background, indicating it is selected or being edited. A tooltip for this element states: "Required z/OS Connect API Requester".

A red oval highlights the status bar message "Press Ctrl+space for content assist.", which is a common feature in many IDEs for providing code completion suggestions.

A modal dialog box titled "Preserve JSON payload character format" is open, showing two options: "true" and "false (default)". This dialog is part of the "Partial reader group" configuration.

Another smaller modal dialog box titled "Partial reader group" is partially visible in the background.

The left sidebar contains a navigation tree with various sections such as "Feature Manager", "z/OS Connect Manager", "z/OS Logging", "z/OS Connect policy name", "Cross-Origin Resource Sharing", "HTTP Endpoint", "Configuration Management", "z/OS Connect APIs", "z/OS Connect Services", and "Application Monitoring".



Use the restConnector-2.0 feature to see real time configuration details

A secure, REST administrative connector that enables remote access from a Java client or Web browser (GET only) or directly through an HTTPS call to the current runtime configuration.

Server Config

restConnector.xml

Read only

Close

Design Source

```
1<?xml version="1.0" encoding="UTF-8"?>
2
3<server description="REST Connector">
4  <featureManager>
5    <feature>restConnector-2.0</feature>
6  </featureManager>
7
8</server>
9
```

URI Path is the concatenation of the path /ibm/api/config with the server XML configuration element and any optional query strings.

<https://mpz3.washington.ibm.com:9443/ibm/api/config/jmsQueue>
https://mpz3.washington.ibm.com:9443/ibm/api/config/zosconnect_cicsIpicConnection?port=1491
https://mpz3.washington.ibm.com:9443/ibm/api/config/zosconnect_zosConnectServiceRestClientConnection
https://mpz3.washington.ibm.com:9443/ibm/api/config/zosconnect_cicsIpicConnection?id=miniloan
<https://mpz3.washington.ibm.com:9443/ibm/api/config/safCredentials>
<https://mpz3.washington.ibm.com:9443/ibm/api/config/connectionFactory>
https://mpz3.washington.ibm.com:9443/ibm/api/config/zosconnect_zosConnectManager
<https://mpz3.washington.ibm.com:9443/ibm/api/config/keyStore>
<https://mpz3.washington.ibm.com:9443/ibm/api/config/ssl>
<https://mpz3.washington.ibm.com:9443/ibm/api/config/sslDefault>
https://mpz3.washington.ibm.com:9443/ibm/api/config/zosconnect_zosConnectManager
https://mpz3.washington.ibm.com:9443/ibm/api/config/zosconnect_zosConnectAPIs
https://mpz3.washington.ibm.com:9443/ibm/api/config/zosconnect_services
https://mpz3.washington.ibm.com:9443/ibm/api/config/zosconnect_apiRequesters

```
RDEFINE EJBROLE BBGZDFLT.com.ibm.ws.management.security.resource.Administrator OWNER(SYS1) UACC(NONE)
RDEFINE EJBROLE BBGZDFLT.com.ibm.ws.management.security.resource.Reader OWNER(SYS1) UACC(NONE)
RDEFINE EJBROLE BBGZDFLT.com.ibm.ws.management.security.resource.allAuthenticatedUsers OWNER(SYS1) UACC(NONE)
PERMIT BBGZDFLT.com.ibm.ws.management.security.resource.Administrator CLASS(EJBROLE) ID(ZCEEUSRS) ACCESS(READ)
PERMIT BBGZDFLT.com.ibm.ws.management.security.resource.Reader CLASS(EJBROLE) ID(ZCEEUSRS) ACCESS(READ)
PERMIT BBGZDFLT.com.ibm.ws.management.security.resource.allAuthenticatedUsers CLASS(EJBROLE) ID(ZCEEUSRS)
ACCESS(READ)
SETR RACLIST(EJBROLE) REFRESH
```



restConnector-2.0 feature

https://mpz3.washington.ibm.com:9443/ibm/api/config/zosconnect_cicsIpicConnection?port=1491

The screenshot shows a web browser window displaying a JSON array of configuration elements. The URL in the address bar is https://mpz3.washington.ibm.com:9443/ibm/api/config/zosconnect_cicsIpicConnection?port=1491. The JSON data lists four connection configurations:

```
[{"configElementName": "zosconnect_cicsIpicConnection", "uid": "catalog", "id": "catalog", "connectionTimeout:": 30000, "heartbeatInterval": 30000, "host": "wg31.washington.ibm.com", "port": 1491, "sharedPort": false, "transidUsage": "EIB_AND_MIRROR"}, {"configElementName": "zosconnect_cicsIpicConnection", "uid": "cscvinc", "id": "cscvinc", "connectionTimeout": 30000, "heartbeatInterval": 30000, "host": "wg31.washington.ibm.com", "port": 1491, "sharedPort": false, "transidUsage": "EIB_AND_MIRROR"}, {"configElementName": "zosconnect_cicsIpicConnection", "uid": "minilcan1", "id": "minilcan1", "connectionTimeout": 30000, "heartbeatInterval": 30000, "host": "wg31.washington.ibm.com", "port": 1491, "sharedPort": false, "transidUsage": "EIB_AND_MIRROR"}, {"configElementName": "zosconnect_cicsIpicConnection", "uid": "minilcan", "id": "minilcan", "connectionTimeout": 30000, "heartbeatInterval": 30000, "host": "wg31.washington.ibm.com", "port": 1491, "sharedPort": false, "transidUsage": "EIB_AND_MIRROR"}]
```



Use the **apiDiscovery-1.0** or **OpenAPI-3.0** features to execute RESTful APIs directly*

The screenshot shows a browser window titled "IBM REST API Documentation". The address bar indicates the URL is <https://mpz3.washington.ibm.com:9443/api/explorer/#/cscvinc>. The main content area is titled "Liberty REST APIs" and subtitle "Discover REST APIs available within Liberty". It lists several API endpoints under the "cscvinc" category:

Method	Endpoint	Operations
POST	/cscvinc/employee	Show/Hide List Operations Expand Operations
DELETE	/cscvinc/employee/{employee}	Show/Hide List Operations Expand Operations
GET	/cscvinc/employee/{employee}	Show/Hide List Operations Expand Operations
PUT	/cscvinc/employee/{employee}	Show/Hide List Operations Expand Operations

Below this, other categories listed include "db2employee", "filemgr", "imsPhoneBook", "jwtIvpDemoApi", "miniloancics", "mqapi", and "phonebook", each with their own "Show/Hide", "List Operations", and "Expand Operations" links.

*V3.0.48



Provide remote access to configuration/log information

The image displays three browser windows illustrating remote access to server configuration and log files:

- Top Left Window:** Shows the XML configuration file for a new server. The content includes various include statements for security, keying, group access, shared, OAuth, and admin center XML files.
- Top Right Window:** Shows the XML configuration for a web application named "serverConfig". It defines a context root of "/server/config", enables file serving and directory browsing, and specifies an extended document root at "\${server.config.dir}".
- Bottom Left Window:** Shows a log file titled "messages.log" from a Java runtime environment. The log entries provide details about the server's configuration, such as product version (WAS FOR Z/OS 20.0.0.6), installation directory, and Java runtime environment.
- Bottom Middle Window:** Shows a log file titled "trace.log" from a Java runtime environment. The log entries show trace events, including a trace state change from "all:SSL=Saf" to "all:SSL=SSLC", and various dependency manager and thread identity bundle file wrapper entries.



Provide remote access to z/OS Connect OPENAPI 2 archives files

Name	Last Modified	Size	Description
apis	Fri Feb 19 13:46:13 GMT 2021	-	Directory
services	Sat Feb 20 20:54:41 GMT 2021	-	Directory
apiRequesters	Wed Feb 07 17:59:04 GMT 2018	-	Directory
rules	Tue Jan 26 20:34:05 GMT 2021	-	Directory

Name	Last Modified	Size	Description
cscvincDeleteService.sar	Thu Feb 18 18:02:19 GMT 2021	4362	File
cscvincInsertService.sar	Thu Feb 18 18:02:19 GMT 2021	4491	File
cscvincSelectService.sar	Thu Feb 18 18:02:19 GMT 2021	4590	File

Opening cscvincSelectService.sar

You have chosen to open:
cscvincSelectService.sar
which is: SAR file (4.5 KB)
from: https://wg31.washington.ibm.com:9453

What should Firefox do with this file?

Open with Applications\WINZIP32.EXE (default)

Save File

OK Cancel

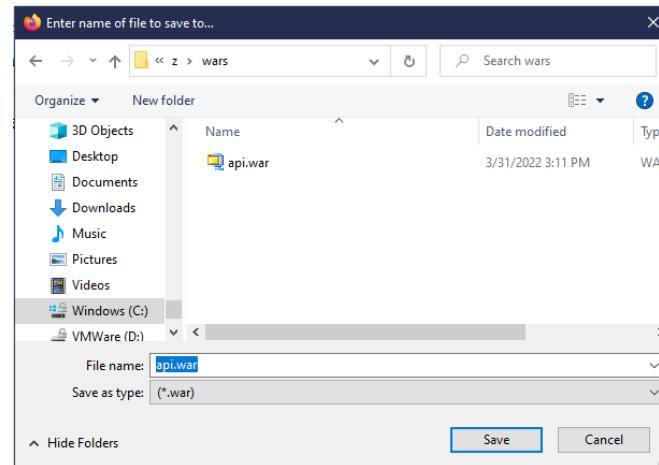

```
<webApplication
  id="resources-location" name="resources"
  location="${server.config.dir}/resources/zosconnect">
  <web-ext context-root="/resources/zosConnect"
    enable-file-serving="true"
    enable-directory-browsing="true">
    <file-serving-attribute name="extendedDocumentRoot"
      value="${server.config.dir}/resources/zosconnect"/>
  </web-ext>
</webApplication>
```

Provide remote access to z/OS Connect Designer OPENAPI 3 web archives files

The screenshot shows a web browser window with the URL <https://localhost:9445/dropins/>. The title bar says "Index of /dropins/". The page content is titled "Index of /dropins/" and contains a table with two rows:

Name	Last Modified	Size	Description
EmployeesApi.war	Tue May 03 22:36:07 UTC 2022	26394	File
api.war	Wed May 04 12:33:45 UTC 2022	15227	File

```
<webApplication id="resources-location" name="resources"
location="/opt/ibm/wlp/usr/servers/defaultServer/dropins">
<web-ext context-root="dropins"
enable-file-serving="true" enable-directory-browsing="true">
<file-servering-attribute name="extendDocumentRoot"
value="/opt/ibm/wlp/usr/servers/defaultServer/dropins" />
</web-ext>
</webApplication> >
```





Liberty MVS Commands

F BAQSTRT,CACHE,CLEAR,AUTH

Clears all users that are cached in the Liberty authentication cache

F BAQSTRT,REFRESH,CONFIG

Process pending configuration updates. Configuration processing applies to the server.xml file, any files it includes

F BAQSTRT,REFRESH,APPS

Process pending application updates. ([Applicable to OpenAPI 3 servers only](#))

F BAQSTRT,REFRESH,KEYSTORE

Use the command to refresh the keystore instorage profiles for the server.

F BAQSTRT,REFRESH,KEYSTORE, ID=OutboundKeyRing

To refresh a specific keystore defined in the server XML with ID=OutboundKeyRing.

F BAQSTRT,CACHE,CLEAR,AUTH

Clears all users that are cached in the Liberty authentication cache.

F BAQSTRT,PAUSE

To pause the server

F BAQSTRT,STATUS

To display the current status of a server

F BAQSTRT,RESUME

To resume the server

For more details, see URL <https://www.ibm.com/docs/en/was-liberty/zos?topic=zos-modify-commands>



Liberty MVS Angel Commands

F BAQZANGL,DISPLAY,SERVERS

Displays a list of servers currently connected to the angel

F BAQZANGL,DISPLAY,SERVERS,PID

Displays a list of servers currently connected to the angel code along with the server's PIDs.

```
CWWKB0067I ANGEL DISPLAY OF ACTIVE SERVERS
CWWKB0080I ACTIVE SERVER ASID 4d JOBNAME ZCEEAPIR PID 16777398
CWWKB0080I ACTIVE SERVER ASID 4b JOBNAME ZCEEDVM PID 50331780
CWWKB0080I ACTIVE SERVER ASID 4f JOBNAME WLPRPSRV PID 138
CWWKB0080I ACTIVE SERVER ASID 4a JOBNAME ZCEESRVR PID 50331815
CWWKB0080I ACTIVE SERVER ASID 50 JOBNAME ZCEEOPID PID 33554605
CWWKB0080I ACTIVE SERVER ASID 4c JOBNAME ZCEEHATS PID 143
CWWKB0080I ACTIVE SERVER ASID 4e JOBNAME WLPOPSRV PID 33554565
CWWKB0080I ACTIVE SERVER ASID 58 JOBNAME MQWEBS PID 152
```

F BAQZANGL,VERSION

Displays the version level of the angel



z/OS Connect MVS Commands (OpenAPI 2)

```
<feature>zosconnect:zosConnectCommands-1.0</feature>
```

F BAQSTRT,ZCON,REFRESH

All updated z/OS Connect artifacts (APIs, services, and API Requesters) are reloaded.

F BAQSTRT,ZCON,CLEARTOKENCACHE

Clears all OAuth 2.0 access tokens and JWTs from the cache. The token cache is only applicable for OAuth 2.0 access tokens and JWTs that were generated either locally or by an external authentication server, when invoking API requesters.

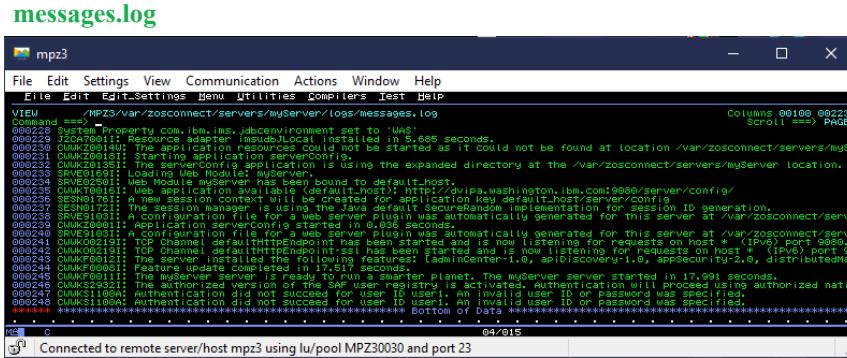
F BAQSTRT,ZCON,CLEARSAFCACHE

Clears the SAF cache. The SAF cache contains SAF user IDs and any associated RACF groups in which the user ID resides. The SAF cache is only applicable to API requester, and only when ID assertion is enabled.

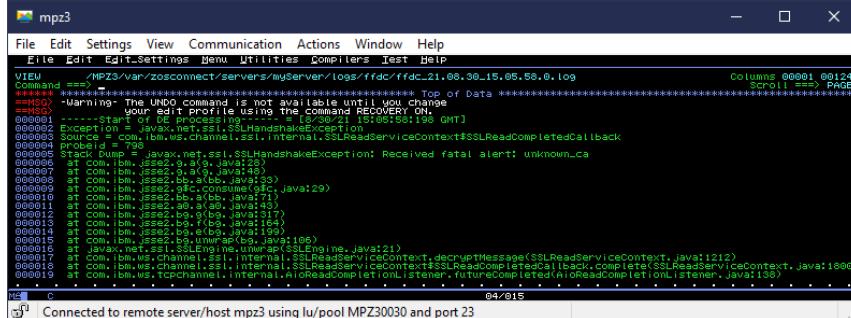
F BAQSTRT,REFRESH,APPS

Where do I look when things go wrong?

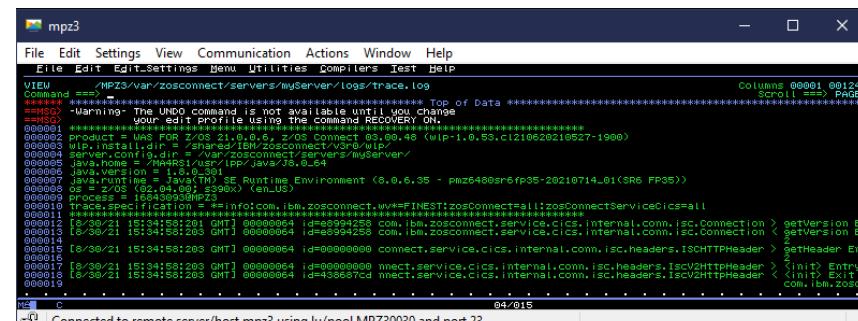
Where to find information when a problem occurs.



First Failure Data Collection (FFDC) dumps



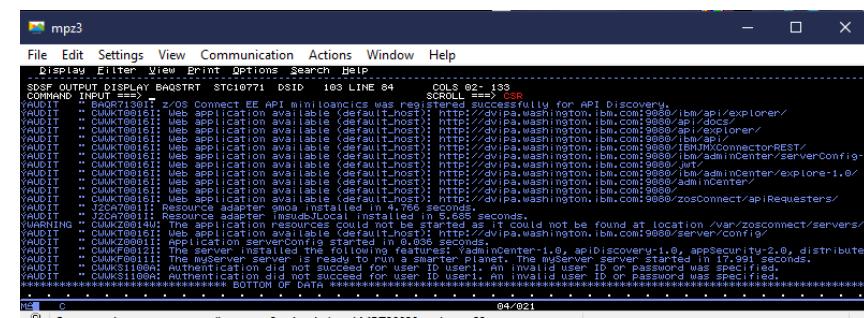
trace.out



mitchj@us.ibm.com



STC STDOUT DD



© 2017, 2023 IBM Corporation
Slide 205



Issues and problems can be categorized

- First realize that actual products problems do occur, but they are rare. In my experience most problems and issues can be resolved with a little investigation and some analysis. I have found that most problems and issues will fall in these categories.

- **Basic Security issues**
 - Insufficient access to local SAF resources, e.g., APPL, EJBROLE, SERVER resources
 - Security issues related to XML configuration elements, safCredentials, sslDefault, keystore, etc.

- **Advanced Security issues**
 - Key ring access, e.g., FACILITY resources IRR.DIGTCERT or RDATALIB or IDIDMAP resources.
 - Key ring contents, e.g., missing certificates, key usage, personal and certificate authorities, private keys versus public keys.
 - Incorrect use of certificates in a TLS handshakes versus certificates used for token validation.

- **z/OS Connect XML Configuration issues**
 - Missing or misspelled configuration attributes (remember the Liberty XML parser is too forgiving)

- **External resource Issues**
 - Service provider configuration issues.
 - Timeouts
 - Network Firewalls
 - Resource Security
 - Other resource errors

Remember external symptoms will overlap. But the use of rigor in setting configuration standards and following a process in problem isolation/determination process will help reduce the impact of problems and issues.



messages.log - The anatomy of a message in the messages.log file

```
*****
product = WAS FOR Z/OS 21.0.0.6, z/OS Connect 03.00.48 (wlp-1.0.53.c1210620210527-1900)
wlp.install.dir = /shared/IBM/zosconnect/v3r0/wlp/
server.config.dir = /var/zosconnect/servers/zceepid/
java.home = /MA4RS1/usr/lpp/java/J8.0_64
java.version = 1.8.0_301
java.runtime = Java(TM) SE Runtime Environment (8.0.6.35 - pmz6480sr6fp35-20210714_01(SR6 FP35))
os = z/OS (02.04.00; s390x) (en_US)
process = 16843186@MPZ3
*****
[9/3/21 13:38:02:831 GMT] 00000013 com.ibm.ws.kernel.launch.internal.FrameworkManager
[9/3/21 13:38:04:439 GMT] 0000001f com.ibm.ws.config.xml.internal.XMLConfigParser
[9/3/21 13:38:04:466 GMT] 0000001f com.ibm.ws.config.xml.internal.XMLConfigParser
[9/3/21 13:38:04:470 GMT] 0000001f com.ibm.ws.config.xml.internal.XMLConfigParser
[9/3/21 13:38:04:473 GMT] 0000001f com.ibm.ws.config.xml.internal.XMLConfigParser
[9/3/21 13:38:04:476 GMT] 0000001f com.ibm.ws.config.xml.internal.XMLConfigParser
[9/3/21 13:38:04:481 GMT] 0000001f com.ibm.ws.config.xml.internal.XMLConfigParser
[9/3/21 13:38:04:610 GMT] 00000021 com.ibm.ws.zos.core.internal.NativeServiceTracker
[9/3/21 13:38:04:612 GMT] 00000021 com.ibm.ws.zos.core.internal.NativeServiceTracker
[9/3/21 13:38:04:628 GMT] 00000021 com.ibm.ws.zos.core.internal.NativeServiceTracker
[9/3/21 13:38:04:679 GMT] 00000021 com.ibm.ws.zos.core.internal.NativeServiceTracker
[9/3/21 13:38:04:680 GMT] 00000021 com.ibm.ws.zos.core.internal.NativeServiceTracker
[9/3/21 13:38:04:680 GMT] 00000021 com.ibm.ws.zos.core.internal.NativeServiceTracker
-----
[9/3/21 13:38:42:347 GMT] 00000040 om.ibm.ws.app.manager.rar.internal.RARApplicationHandlerImpl
[9/3/21 13:38:42:419 GMT] 0000003e com.ibm.ws.jmx.connector.server.rest.RESTAppListener
[9/3/21 13:38:42:422 GMT] 0000003e com.ibm.ws.jmx.connector.server.rest.RESTAppListener
[9/3/21 13:38:42:428 GMT] 0000002c com.ibm.ws.tcpchannel.internal.TCPEndpoint
[9/3/21 13:38:42:431 GMT] 0000002c com.ibm.ws.tcpchannel.internal.TCPEndpoint
[9/3/21 13:38:42:437 GMT] 00000042 com.ibm.ws.webcontainer.osgi.mbeans.PluginGenerator
[9/3/21 13:38:42:489 GMT] 0000002c com.ibm.ws.kernel.feature.internal.FeatureManager
[9/3/21 13:38:42:490 GMT] 0000002c com.ibm.ws.kernel.feature.internal.FeatureManager
[9/3/21 13:38:42:490 GMT] 0000002c com.ibm.ws.kernel.feature.internal.FeatureManager
[9/3/21 13:41:31:640 GMT] 00000045 .security openidconnect.client.internal.OidcClientConfigImpl
[9/3/21 13:41:31:691 GMT] 00000045 liberty.authentication.filter.internal.AuthenticationFilterImpl
[9/3/21 13:41:32:824 GMT] 00000053 com.ibm.zosconnect.service.cics.internal.conn.isc.Connection
*****
A CWWKE0001I: The server zceepid has been launched.
A CWWKG0028A: Processing included configuration resource
I CWWKB0125I: This server requested a REGION size of 0KB
I CWWKB0126I: MEMLIMIT=2000. MEMLIMIT CONFIGURATION SOUR
I CWWKB0122I: This server is connected to the default an
I CWWKB0103I: Authorized service group KERNEL is availab
I CWWKB0103I: Authorized service group LOCALCOM is avail
I CWWKB0103I: Authorized service group PRODMGR is availa
----- 148 Line(s) not Displayed
A J2CA7001I: Resource adapter imsudbJLocal installed in
I CWWKX0103I: The JMX REST connector is running and is a
I CWWKX0103I: The JMX REST connector is running and is a
I CWWKO0219I: TCP Channel defaultHttpEndpoint has been s
I CWWKO0219I: TCP Channel defaultHttpEndpoint-ssl has be
I SRVE9103I: A configuration file for a web server plugi
A CWWKF0012I: The server installed the following feature
I CWWKF0008I: Feature update completed in 37.484 seconds
A CWWKF0011I: The zceepid server is ready to run a smar
I CWWKS1700I: OpenID Connect client ATS configuration su
I CWWKS4358I: The authentication filter ATSAuthFilter co
BAQR0680I: CICS connection cscvinc established with 10
```

- **WLP_LOGGING_CONSOLE_FORMAT - SIMPLE** - Use the simple logging format. As of Liberty release 20.0.0.6 (z/OS Connect V3.034), this format writes the messages to STDOUT and STDERR with time stamps included.



Basic security issues – Sometimes the problem is easy to find

The STDOUT may show:

```
ÝAUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified  
ÝAUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified
```

And the messages.log displays:

```
CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.  
CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
```

But the JESMSGGLG and SYSLOG displays:

```
WG31
File Edit Settings View Communication Actions Window Help
Display Filter View Print Options Search Help

SDSF OUTPUT DISPLAY BAQSTRT STC12856 DSID 2 LINE 17 COLS 02- 133
COMMAND INPUT ==> SCROLL ==> PAGE
15.27.30 STC12856 +CWWKZ0001I: Application resources started in 0.086 seconds.
15.27.30 STC12856 +CWWKZ0001I: Application serverConfig started in 0.085 seconds.
15.27.31 STC12856 +CWWKF0011I: The myServer server is ready to run a smarter planet. The 001
001 myServer server started in 36.923 seconds.
15.27.32 STC12856 ICH408I USER(FRED ) GROUP(ATSGRP ) NAME(USER FRED ) 002
002 LOGON/JOB INITIATION - INVALID PASSWORD
15.27.32 STC12856 IRR013I VERIFICATION FAILED. INVALID PASSWORD GIVEN.
15.27.32 STC12856 ICH408I USER(FRED ) GROUP(ATSGRP ) NAME(USER FRED ) 004
004 LOGON/JOB INITIATION - INVALID PASSWORD
15.27.32 STC12856 IRR013I VERIFICATION FAILED. INVALID PASSWORD GIVEN.
1 //BADSTRT JOB MSGLEVEL=1 STC12856
2 //STARTING EXEC BAQSTRT
3 XXBAQSTRT PROC PARMs='myServer --clean'
XX*
4 XX SET ZCONHOME='/usr/lpp/IBM/zosconnect/v3r0'
XX*
5 XXZCON EXEC PGM=BPXBATSL,REGION=0M,MEMLIMIT=8G,
XX PARM='PGM &ZCONHOME./bin/zosconnect run &PARMS.'
IEFC653I SUBSTITUTION JCL - PGM=BPXBATSL,REGION=0M,MEMLIMIT=8G,PARM='PGM
/usr/lpp/IBM/zosconnect/v3r0/bin/zosconnect run myServer --clean'
6 XXSTEPLIB DD DISP=SHR,DSN=MQ91#.SCSQAUTH
7 XX DD DISP=SHR,DSN=MQ91#.SCSQANLE
8 XXSTDERR DD SYSOUT=*,FREE=CLOSE,SPIN=(UNALLOC,1M)
```

MA B 16/074
Connected to remote server/host mpz3 using lu/pool MPZ30023 and port 23



Basic security issues – Sometimes you must dig a little more

The STDOUT may show:

```
ÝAUDIT  .. CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified  
ÝAUDIT  .. CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified
```

But there are no SAF messages in the SYSLOG:

While the messages.log displays a SAF return code and reason code:

```
WG31  
File Edit Settings View Communication Actions Window Help  
File Edit Edit_Settings Menu Utilities Compilers Test Help  
VIEW      /MPZ3/var/zosconnect/servers/myServer/logs/messages.log  
Command ==> -  
Columns 00100 00223  
Scroll ==> PAGE  
000256 SAF return code 0x00000008. RACF return code 0x00000008. RACF reason code 0x00000020.  
000257 CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.  
000258 CWWKS2907E: SAF Service IRRSIA00_CREATE did not succeed because user FRED has insufficient authority to access APPL-ID BBGZD  
000259 CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.  
000260 CWWKS2907E: SAF Service IRRSIA00_CREATE did not succeed because user FRED has insufficient authority to access APPL-ID BBGZD  
000261 CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.  
000262 CWWKS2907E: SAF Service IRRSIA00_CREATE did not succeed because user FRED has insufficient authority to access APPL-ID BBGZD  
000263 CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.  
000264 CWWKS2907E: SAF Service IRRSIA00_CREATE did not succeed because user FRED has insufficient authority to access APPL-ID BBGZD  
000265 CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.  
000266 CWWKS2907E: SAF Service IRRSIA00_CREATE did not succeed because user FRED has insufficient authority to access APPL-ID BBGZD  
000267 CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.  
000268 CWWKS2907E: SAF Service IRRSIA00_CREATE did not succeed because user FRED has insufficient authority to access APPL-ID BBGZD  
000269 CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.  
000270 CWWKS2907E: SAF Service IRRSIA00_CREATE did not succeed because user FRED has insufficient authority to access APPL-ID BBGZD  
000271 CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.  
000272 CWWKS2907E: SAF Service IRRSIA00_CREATE did not succeed because user FRED has insufficient authority to access APPL-ID BBGZD  
000273 CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.  
000274 CWWKS2907E: SAF Service IRRSIA00_CREATE did not succeed because user FRED has insufficient authority to access APPL-ID BBGZD  
000275 CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.  
000276 CWWKS2907E: SAF Service IRRSIA00_CREATE did not succeed because user FRED has insufficient authority to access APPL-ID BBGZD  
000277 CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.  
***** ***** Bottom of Data *****  
A B  
Connected to remote server/host mpz3 using lu/pool MPZ30023 and port 23  
04/015
```

CWWKS2907E: SAF Service IRRSIA00_CREATE did not succeed because user FRED has insufficient authority to access APPL-ID BBGZDFLT. SAF return code 0x00000008. RACF return code 0x00000008. RACF reason code 0x00000020.

mitchj@us.ibm.com

Tech-Tip: And be aware of hex v. decimal in return and reason codes

RACF return code 0x00000008. RACF reason code 0x00000020.

Table 1. initACEE create return codes

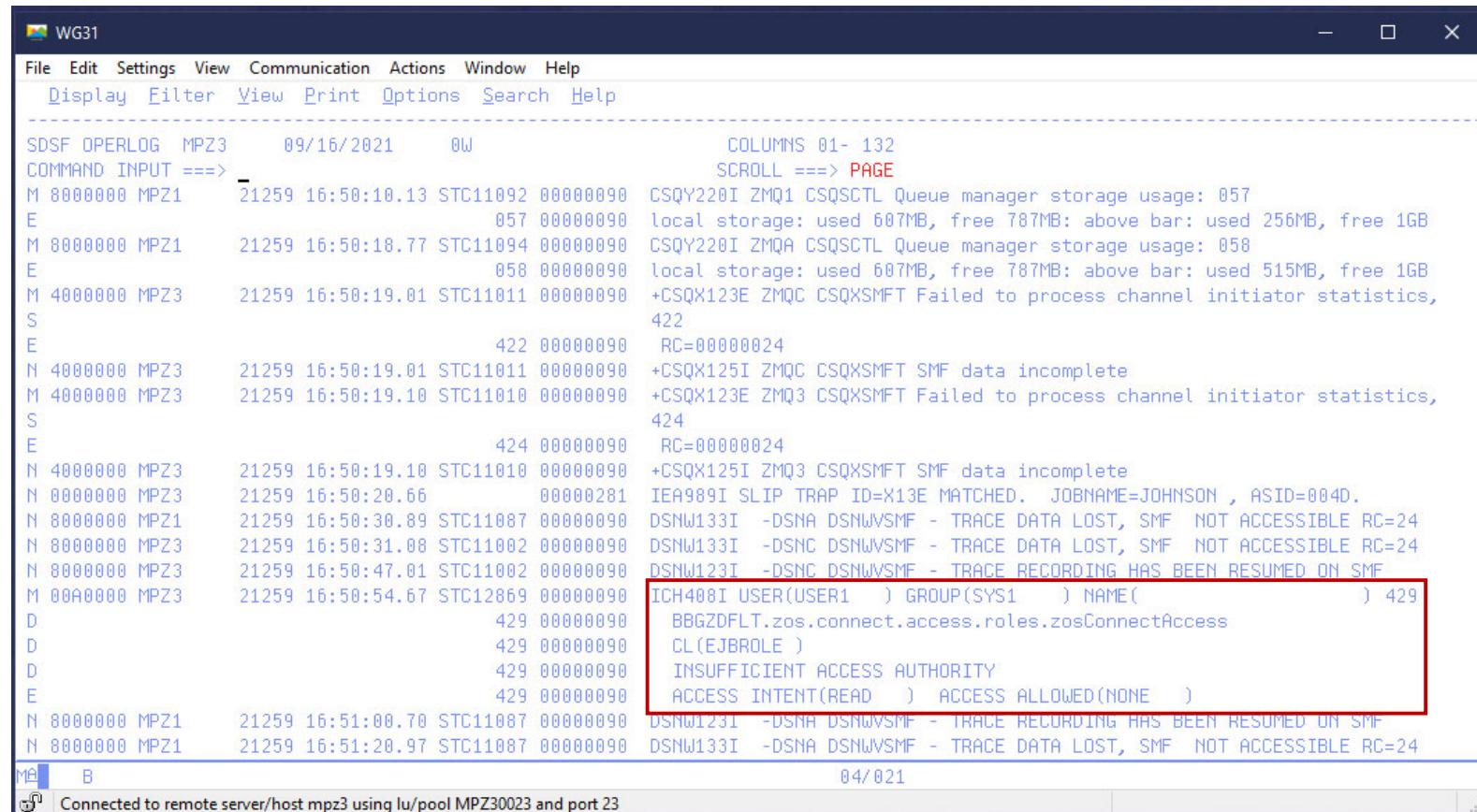
SAF return code	RACF® return code	RACF reason code	Explanation
0	0	0	The service was successful.
4	0	0	RACF is not installed.
8	8	4	Parameter list error occurred.
8	8	8	An internal error occurred during RACF processing.
8	8	12	Recovery environment could not be established.
8	8	16	User ID is not defined to RACF.
8	8	20	Password, Password Phrase or Pass Ticket is not valid.
8	8	24	Password or Password Phrase is expired.
8	8	28	User ID is revoked or user access to group is revoked.
8	8	32	The user does not have appropriate RACF access to either the SECLABEL, SERVAUTH profile, or APPL specified in the parmlist.
8	8	36	Certificate is not valid.
8	8	40	No user ID is defined for this certificate. See Usage Note number 37.
8	8	44	The client security label is not equivalent to the server's security label.
8	8	48	A managed ACEE is requested with a nested RACO in the Envir_In parameter.
8	12	InitUSP reason code	initUSP failed. See initUSP reason codes in Return and reason codes .

Hex '20' = Dec '32'

Root cause – No READ access to APPL resource BBGZDFLT

Basis security issues - Use the SYSLOG/JESMSGGLG output

The SYSLOG shows a ICH408I message:



```

WG31
File Edit Settings View Communication Actions Window Help
Display Filter View Print Options Search Help
SDSF OPERLOG MPZ3 09/16/2021 0W
COMMAND INPUT ===> -
M 8000000 MPZ1 21259 16:50:10.13 STC11092 00000090 CSQY220I ZMQ1 CSQSCTL Queue manager storage usage: 057
E 057 00000090 local storage: used 607MB, free 787MB; above bar: used 256MB, free 1GB
M 8000000 MPZ1 21259 16:50:18.77 STC11094 00000090 CSQY220I ZMQA CSQSCTL Queue manager storage usage: 058
E 058 00000090 local storage: used 607MB, free 787MB; above bar: used 515MB, free 1GB
M 4000000 MPZ3 21259 16:50:19.01 STC11011 00000090 +CSQX123E ZMQC CSQXSMFT Failed to process channel initiator statistics,
S 422
E 422 00000090 RC=00000024
N 4000000 MPZ3 21259 16:50:19.01 STC11011 00000090 +CSQX125I ZMQC CSQXSMFT SMF data incomplete
M 4000000 MPZ3 21259 16:50:19.10 STC11010 00000090 +CSQX123E ZMQ3 CSQXSMFT Failed to process channel initiator statistics,
S 424
E 424 00000090 RC=00000024
N 4000000 MPZ3 21259 16:50:19.10 STC11010 00000090 +CSQX125I ZMQ3 CSQXSMFT SMF data incomplete
N 0000000 MPZ3 21259 16:50:20.66 000000281 IEA989I SLIP TRAP ID=X13E MATCHED. JOBNAME=JOHNSON , ASID=004D.
N 8000000 MPZ1 21259 16:50:30.89 STC11087 00000090 DSNW133I -DSNA DSNWVSMF - TRACE DATA LOST, SMF NOT ACCESSIBLE RC=24
N 8000000 MPZ3 21259 16:50:31.08 STC11002 00000090 DSNW133I -DSNC DSNWVSMF - TRACE DATA LOST, SMF NOT ACCESSIBLE RC=24
N 8000000 MPZ3 21259 16:50:47.01 STC11002 00000090 DSNW123T -DSNC DSNWVSMF - TRACE RECORDING HAS BEEN RESUMED ON SMF
M 00A0000 MPZ3 21259 16:50:54.67 STC12869 00000090 ICH408I USER(USER1 ) GROUP(SYS1 ) NAME( ) 429
D 429 00000090 BBGZDFLT.zos.connect.access.roles.zosConnectAccess
D 429 00000090 CL(EJBROLE )
D 429 00000090 INSUFFICIENT ACCESS AUTHORITY
E 429 00000090 ACCESS INTENT(READ ) ACCESS ALLOWED(NONE )
N 8000000 MPZ1 21259 16:51:00.70 STC11087 00000090 DSNW123I -DSNA DSNWVSMF - TRACE RECORDING HAS BEEN RESUMED ON SMF
N 8000000 MPZ1 21259 16:51:20.97 STC11087 00000090 DSNW133I -DSNA DSNWVSMF - TRACE DATA LOST, SMF NOT ACCESSIBLE RC=24

```

Connected to remote server/host mpz3 using lu/pool MPZ30023 and port 23

Symptom: client see HTTP 403 – Authorization Failed. There were no messages in STDOUT or messages.log locations. Root cause – No READ access to EJBROLE BBGZDFLT.zos.connect.access.roles.zosConnectAccess.



Basic security issues – Sometimes there is misdirection

The STDOUT may show:

WG31

File Edit Settings View Communication Actions Window Help

Display Filter View Print Options Search Help

```
SDSF OUTPUT DISPLAY BAQSTRT STC12844 DSID 103 LINE 98      COLS 02- 133
COMMAND INPUT ===> SCROLL ==> PAGE
AUDIT  " CWWKZ0001I: Application serverConfig started in 4.006 seconds.
AUDIT  " CWWKZ0001I: Application resources started in 4.007 seconds.
AUDIT  " CWWKT0016I: Web application available (default_host): http://dvipa.washington.ibm.com:9080/zosConnect/apiRequesters/
AUDIT  " CWWKT0016I: Web application available (default_host): http://dvipa.washington.ibm.com:9080/
AUDIT  " CWWKF0012I: The server installed the following features: YadminCenter-1.0, apiDiscovery-1.0, appSecurity-2.0, distributed
AUDIT  " CWWKF0011I: The myServer server is ready to run a smarter planet. The myServer server started in 66.646 seconds.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
AUDIT  " CWWKS1100A: Authentication did not succeed for user ID FRED. An invalid user ID or password was specified.
***** BOTTOM OF DATA *****
```

M A B 04/021

Connected to remote server/host mpz3 using lu/pool MPZ30019 and port 23

Basic security issues - SYSLOG/JESMSGGLG output (even more misdirection)



```

WG31

File Edit Settings View Communication Actions Window Help
Display Filter View Print Options Search Help
-----
SDSF OUTPUT DISPLAY BAQSTRT STC12862 DSID      2 LINE 0      COLS 02- 133
COMMAND INPUT ==> SCROLL ==> PAGE
***** TOP OF DATA *****
J E S 2   J O B   L O G   --   S Y S T E M   M P Z 3   --   N O D E   W S C 1 0

16.31.55 STC12862 ---- THURSDAY, 16 SEP 2021 ----
16.31.55 STC12862 IEF695I START BAQSTRT WITH JOBNAME BAQSTRT IS ASSIGNED TO USER LIBSERV , GROUP LIBGRP
16.31.55 STC12862 $HASP373 BAQSTRT STARTED
16.32.03 STC12862 +CLWJKE0001I: The server myServer has been launched.
16.32.20 STC12862 BPXMF023I (LIBSERV) 282
    282     GMODIG7777I: IMS service provider (20210816-0926) for z/OS Connect
    282     Enterprise Edition initialized successfully.
16.32.50 STC12862 +CLWJKZ0001I: Application resources started in 14.912 seconds.
16.32.50 STC12862 +CLWJKZ0001I: Application serverConfig started in 14.910 seconds.
16.32.55 STC12862 +CLWJKF0011I: The myServer server is ready to run a smarter planet. The 285
    285     myServer server started in 51.809 seconds
16.43.25 STC12862 BPXP015I HFS PROGRAM /usr/lib/java_runtime/libifaedjreg64.so IS NOT MARKED PROGRAM CONTROLLED.
16.43.25 STC12862 BPXP014I ENVIRONMENT MUST BE CONTROLLED FOR DAEMON (BPX.DAEMON) PROCESSING.
16.43.25 STC12862 BPXP015I HFS PROGRAM /usr/lib/java_runtime/libifaedjreg64.so IS NOT MARKED PROGRAM CONTROLLED.
16.43.25 STC12862 BPXP014I ENVIRONMENT MUST BE CONTROLLED FOR DAEMON (BPX.DAEMON) PROCESSING.
16.43.26 STC12862 BPXP015I HFS PROGRAM /usr/lib/java_runtime/libifaedjreg64.so IS NOT MARKED PROGRAM CONTROLLED.
16.43.26 STC12862 BPXP014I ENVIRONMENT MUST BE CONTROLLED FOR DAEMON (BPX.DAEMON) PROCESSING.
16.43.26 STC12862 BPXP015I HFS PROGRAM /usr/lib/java_runtime/libifaedjreg64.so IS NOT MARKED PROGRAM CONTROLLED.
16.43.26 STC12862 BPXP014I ENVIRONMENT MUST BE CONTROLLED FOR DAEMON (BPX.DAEMON) PROCESSING.
16.43.26 STC12862 BPXP015I HFS PROGRAM /usr/lib/java_runtime/libifaedjreg64.so IS NOT MARKED PROGRAM CONTROLLED.

MA B          04/021
Connected to remote server/host mpz3 using lu/pool MPZ30023 and port 23

```

Symptom: Client unable to connect. STDOUT contains message *CWWKS1100A: Authentication did not succeed for user ID user1. An invalid user ID or password was specified.*



Basic security issues - SYSLOG/JESMSGGLG output (even more misdirection)

There is no need to set the extended protection attribute for this Java shared object executable.

The root cause was that the angel was not active.

```
VIEW      /MPZ3/var/zosconnect/servers/myServer/logs/messages.log          Columns 00100 00223
Command ==>
000021 CUWKG0028A: Processing included configuration resource: /var/zosconnect/servers/myServer/includes/shared.xml
000022 CUWKG0028A: Processing included configuration resource: /var/zosconnect/servers/myServer/includes/oauth.xml
000023 CUWKG0028A: Processing included configuration resource: /var/zosconnect/servers/myServer/includes/audit.xml
000024 CUWKG0028A: Processing included configuration resource: /var/zosconnect/servers/myServer/includes/mq.xml
000025 CUWKG0028A: Processing included configuration resource: /var/zosconnect/servers/myServer/includes/db2.xml
000026 CUWKG0028A: Processing included configuration resource: /var/zosconnect/servers/myServer/includes/wlm.xml
000027 CUWKG0028A: Processing included configuration resource: /var/zosconnect/servers/myServer/includes/restConnector.xml
000028 CUWKG0028A: Processing included configuration resource: /var/zosconnect/servers/myServer/includes/smf.xml
000029 CUWKG0028A: Processing included configuration resource: /var/zosconnect/servers/myServer/includes/adminCenter.xml
000030 CUWKB0125I: This server requested a REGION size of 8KB. The below-the-line storage limit is 8MB and the above-the-line stor
000031 CUWKB0126I: MEM1 TMIT=2000... MEM1 TMIT CONFIGURATION SOURCE=TCI
000032 CUWKB0101I: The angel process is not available. No authorized services will be loaded. The reason code is 4.
000033 CUWKB0104I: Authorized service group KERNEL is not available.
000034 CUWKB0104I: Authorized service group LOCALCOM is not available.
000035 CUWKB0104I: Authorized service group PRODMGR is not available.
000036 CUWKB0104I: Authorized service group SAFCRED is not available.
000037 CUWKB0104I: Authorized service group TXRRS is not available.
000038 CUWKB0104I: Authorized service group WOLA is not available.
000039 CUWKB0104I: Authorized service group ZOSAIO is not available.
000040 CUWKB0104I: Authorized service group ZOSDUMP is not available.
000041 CUWKB0104I: Authorized service group ZOSLWM is not available.
000042 CUWKB0104I: Authorized service group CLIENT.WOLA is not available.
000043 CUWKB0108I: IBM Corp product z/OS Connect version 03.00 successfully registered with z/OS.
MA      B                                         14/809
Connected to remote server/host mpz3 using lu/pool MPZ30023 and port 23
```



External resource issues (HTTP 500)

The client sees:

```
HTTP/1.1 500 Internal Server Error
```

The STDOUT may show:

```
ÝWARNING " BAQR0429W: API db2employee encountered an error while processing a request under URL  
https://mpz3.washington.ibm.com:9443/db2/employee/948478.
```

While the messages.log display

```
[9/16/21 21:00:55:811 GMT] 00000051 com.ibm.zosconnect.service.cics.internal.conn.ISCECIRequest E BAQR0657E: Transaction  
abend MIJO occurred in CICS while using CICS connection cscvinc and service cscvincDeleteService.  
[9/16/21 21:00:55:815 GMT] 00000051 com.ibm.zosconnect.internal.web.ServiceProxyServlet W BAQR0429W: API cscvinc  
encountered an error while processing a request under URL https://mpz3.washington.ibm.com:9443/cscvinc/employee/948478.
```

The STDOUT may show:

```
ÝWARNING " BAQR0429W: API db2employee encountered an error while processing a request under URL  
https://mpz3.washington.ibm.com:9443/db2/employee/948478.
```

The messages.log displays:

```
[9/14/21 20:04:59:776 GMT] 00000048 osconnect.service.client.rest.internal.RestClientServiceImpl E BAQR0558E: The remote  
service invocation failed with [9/14/21 20:04:59:776 GMT] 00000048  
osconnect.service.client.rest.internal.RestClientServiceImpl E BAQR0558E: The remote service invocation failed with failed  
due to SQLCODE=-204 SQLSTATE=42704, USER1.EMPLOYEE IS AN UNDEFINED NAME. Error Location:DSNLJACC:35"}
```



Tech-Tip: An HTTP 500 shortcut – look elsewhere

A HTTP status code 500 occurs when a failure occurred at an external endpoint. It does not matter if the external endpoint is a z/OS resources or a REST API provider, or an authorization server, etc.

The details of the failure may not be provided **directly** to z/OS Connect, just the fact that a failure has occurred. The failure could be a security issue, an abend or something entirely. z/OS Connect may or may not have directly access to any details of the failure (it depends on the service provider). It does not mean the details do not exist; the details are just readily available.

The shortcut to identify the issue is review the messages in the messages.log and check to see if there is corresponding FFDC (first failure data collection) dump.



What is a Java stack trace?

```
[9/6/21 22:51:19:981 GMT] 00000039 com.ibm.ejs.j2c.ConnectionEventListener
A J2CA0056I: The Connection Manager received
a fatal connection error from the Resource Adapter for resource null. The exception is: javax.resource.spi.EISSystemException: ICO0001E:
com.ibm.connector2.ims.ico.IMSTCIPManagedConnection@c341a0aa.processOutputOTMAMsg(Connection, InteractionSpec, Record, Record) error. IMS
Connect returned an error: RETCODE=[4], REASONCODE=[NFNDDST] [Datastore not found. ]
at com.ibm.connector2.ims.ico.IMSManagedConnection.processOutputOTMAMsg(IMSManagedConnection.java:4042)
at com.ibm.connector2.ims.ico.IMSTCIPManagedConnection.callSendRecv(IMSTCIPManagedConnection.java:241)
at com.ibm.connector2.ims.ico.IMSManagedConnection.call(IMSManagedConnection.java:1625)
at com.ibm.connector2.ims.ico.IMSConnection.call(IMSConnection.java:213)
at com.ibm.connector2.ims.ico.IMSInteraction.execute(IMSInteraction.java:586)
at com.ibm.ims.gateway.services.IMSGatewayServiceImpl.executeTransServiceInputTMRA(Unknown Source)
at com.ibm.ims.gateway.services.IMSGatewayServiceImpl.invokeTransactionService(Unknown Source)
at com.ibm.ims.gateway.services.IMSGatewayServiceImpl.invoke(Unknown Source)
at com.ibm.ims.zconnect.provider.clients.GatewayServiceClient.doPost(Unknown Source)
at com.ibm.ims.zconnect.provider.clients.IMSClient.doInvoke(Unknown Source)
at com.ibm.ims.gateway.config.services.IMSZServiceHandlerImpl.invoke(Unknown Source)
at com.ibm.ims.gateway.config.services.IMSZServiceImpl.invoke(Unknown Source)
at com.ibm.zosconnect.internal.ZosConnectServiceImpl.apiInvoke(Unknown Source)
at com.ibm.zosconnect.internal.ServiceManagerImpl.invoke(Unknown Source)
at com.ibm.zosconnect.internal.ApiManagerImpl.invokeApi(Unknown Source)
at com.ibm.zosconnect.internal.web.ServiceProxyServlet$3.run(Unknown Source)
at com.ibm.ws.webcontainer.async.ServiceWrapper.wrapAndRun(ServiceWrapper.java:236)
at com.ibm.ws.webcontainer.async.ContextWrapper.run(ContextWrapper.java:28)
at com.ibm.ws.webcontainer.async.WrapperRunnableImpl.run(WrapperRunnableImpl.java:89)
at com.ibm.ws.threading.internal.ExecutorServiceImpl$RunnableWrapper.run(ExecutorServiceImpl.java:238)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1160)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:635)
at java.lang.Thread.run(Thread.java:825)
```

A J2CA0056I: The Connection Manager received
a fatal connection error from the Resource Adapter for resource null. The exception is: javax.resource.spi.EISSystemException: ICO0001E:
com.ibm.connector2.ims.ico.IMSTCIPManagedConnection@c341a0aa.processOutputOTMAMsg(Connection, InteractionSpec, Record, Record) error. IMS
Connect returned an error: RETCODE=[4], REASONCODE=[NFNDDST] [Datastore not found.]

IMS service provider classes
z/OS Connect Java classes

A Google search of ICO00001E returned an explanation at URL: <https://www.ibm.com/docs/en/ims/13.1.0?topic=exceptions-ico0001e>

Root cause – Datastore mistyped in the interaction configuration

First Failure Data Collection (FFDC)



```
-----Start of DE processing----- = [9/7/21 14:19:29:291 GMT]
Exception = com.ibm.msg.client.jms.DetailedIllegalStateException
Source = com.ibm.zosconnect.service.mq.OneWayMQServiceInvocation
probeid = 0004
Stack Dump = com.ibm.msg.client.jms.DetailedIllegalStateException: JMSWMQ2002: Failed to get a message from destination 'ZCONN2.DEFAULT.MQZCEE.QUEUE'.
IBM MQ classes for JMS attempted to perform an MQGET; however IBM MQ reported an error.
Use the linked exception to determine the cause of this error.
at com.ibm.msg.client.wmq.common.internal.Reason.reasonToException(Reason.java:489)
at com.ibm.msg.client.wmq.common.internal.Reason.createException(Reason.java:215)
.
.
.
at com.ibm.zosconnect.service.mq.MQService.invoke(Unknown Source)
at com.ibm.zosconnect.internal.ZosConnectServiceImpl.apiInvoke(Unknown Source)
at com.ibm.zosconnect.internal.ServiceManagerImpl.invoke(Unknown Source)
at com.ibm.zosconnect.internal.ApiManagerImpl.invokeApi(Unknown Source)
at com.ibm.zosconnect.internal.web.ServiceProxyServlet$3.run(Unknown Source)
at com.ibm.ws.webcontainer.async.ServiceWrapper.wrapAndRun(ServiceWrapper.java:236)
at com.ibm.ws.webcontainer.async.ContextWrapper.run(ContextWrapper.java:28)
at com.ibm.ws.webcontainer.async.WrapperRunnableImpl.run(WrapperRunnableImpl.java:89)
at com.ibm.ws.threading.internal.ExecutorServiceImpl$RunnableWrapper.run(ExecutorServiceImpl.java:238)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1160)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:635)
at java.lang.Thread.run(Thread.java:825)
Caused by: com.ibm.mq.MQException: JMSCMQ0001: IBM MQ call failed with compcode '2' ('MQCC_FAILED') reason '2016' ('MQRC_GET_INHIBITED').
at com.ibm.msg.client.wmq.common.internal.Reason.createException(Reason.java:203)
... 25 more
```

MQ service provider classes

Root cause – Queue was configured to disable the MQPUT request

The FFDC dump is more than just a Java stack trace



z/OS Connect Java classes

```
-----Start of DE processing----- = [9/7/21 20:26:12:394 GMT]
Exception = com.ibm.zosconnect.endpoint.connection.TokenConfigException
Source = com.ibm.zosconnect.endpoint.connection.internal.OAuthConfigImpl
probeid = 265
Stack Dump = com.ibm.zosconnect.endpoint.connection.TokenConfigException: BAQR1006E: An error occurred when z/OS Connect EE attempted to
access the authentication/authorization server. Error: javax.net.ssl.SSLHandshakeException: SSLHandshakeException invoking
https://wg31.washington.ibm.com:26213/oidc/endpoint/OP/token: com.ibm.jsse2.util.j: PKIX path building failed:
com.ibm.security.cert.IBMCertPathBuilderException: unable to find valid certification path to requested target
at com.ibm.zosconnect.endpoint.connection.internal.OAuthConfigImpl.requestAuthorizationServer(Unknown Source)
at com.ibm.zosconnect.endpoint.connection.internal.OAuthConfigImpl.getAuthData(Unknown Source)
at com.ibm.zosconnect.apirequester.internal.restclient.RestClientImpl.handleAuthConfig(Unknown Source)
at com.ibm.zosconnect.apirequester.internal.restclient.RestClientImpl.invoke(Unknown Source)
at com.ibm.zosconnect.apirequester.internal.ARInvokeHandler.handle(Unknown Source)
at com.ibm.zosconnect.apirequester.internal.ApiRequesterManagerImpl.invoke(Unknown Source)
at com.ibm.zosconnect.apirequester.internal.proxy.ApiRequesterManagerProxyImpl$1.run(Unknown Source)
.
.
Dump of callerThis
Object type = com.ibm.zosconnect.endpoint.connection.internal.OAuthConfigImpl
copyright_notice = "Licensed Materials - Property of IBM 5655-CE3 (c) Copyright IBM Corp. 2017, 2021 All Rights Reserved
tc = class com.ibm.websphere.ras.TraceComponent@2d85bcc
strings[0] = "TraceComponent[com.ibm.zosconnect.endpoint.connection.internal.OAuthConfigImpl,class
com.ibm.zosconnect.endpoint.connection.internal.OAuthConfigImpl,[zosConnectApiRequesterToken],com.ibm.zosconnect.endpoint
.connection.internal.resources.ZosConnectEndpointConnection,null]"
CFG_ELEMENT_ID = "id"
CFG_GRANTTYPE = "grantType"
id = "myoAuthConfig"
grantType = "password"
authServer = class com.ibm.zosconnect.endpoint.connection.internal.AuthorizationServerImpl@ed6c1e8c
.
.
sslCertsRef = "OutboundSSLSettings"
connectionTimeout = 30000
receiveTimeout = 60000
id = "myoAuthServer"
```



The FFDC dump for a network issue

```
-----Start of DE processing----- = [6/6/21 14:56:01:242 GMT]
Exception = java.net.UnknownHostException
Source = com.ibm.zosconnect.service.cics.internal.conn.isc.ConnectionManager
probeid = 131
Stack Dump = java.net.UnknownHostException: wg31.washington.ibm.com
at java.net.InetAddress.getAllByName0 (InetAddress.java:1419)
at java.net.InetAddress.getAllByName (InetAddress.java:1323)
at java.net.InetAddress.getAllByName (InetAddress.java:1246)
at java.net.InetAddress.getByName (InetAddress.java:1196)
at com.ibm.zosconnect.service.cics.internal.conn.isc.ConnectionManager.createConnection (Unknown Source)
at com.ibm.zosconnect.service.cics.internal.conn.isc.ConnectionManager.getConnection (Unknown Source)
at com.ibm.zosconnect.service.cics.internal.conn.isc.SessionManager.getNewConversation (Unknown Source)
at com.ibm.zosconnect.service.cics.ServerECIRequest.executeISC (Unknown Source)
at com.ibm.zosconnect.service.cics.ServerECIRequest.execute (Unknown Source)
at com.ibm.zosconnect.service.cics.internal.CicsIpccConnection.flow (Unknown Source)
at com.ibm.zosconnect.service.cics.internal.CicsServiceImpl.flowRequest (Unknown Source)
at com.ibm.zosconnect.service.cics.internal.CicsServiceImpl.invoke (Unknown Source)
at com.ibm.zosconnect.internal.ZosConnectServiceImpl.apiInvoke (Unknown Source)
at com.ibm.zosconnect.internal.ServiceManagerImpl.invoke (Unknown Source)
at com.ibm.zosconnect.internal.ApiManagerImpl.invokeApi (Unknown Source)
```

Base Java classes
z/OS Connect Java classes

Root cause – Host wg31.washington.ibm.com was not configured in the DNS server



Use the messages.log and FFDC log together

The messages.log states a First Failure Data Collection dump of the issues has been created.

```
[9/12/21 14:56:45:613 GMT] 00000045 com.ibm.ws.logging.internal.impl.IncidentImpl           I FFDC1015I: An FFDC Incident has been  
created: "com.ibm.mq.connector.DetailedResourceException: MQJCA1011: Failed to allocate a JMS connection., error code: MQJCA1011 An  
internal error caused an attempt to allocate a connection to fail. See the linked exception for details of the failure.  
com.ibm.ejs.j2c.poolmanager.FreePool.createManagedConnectionWithMCWrapper 199" at ffdc_21.09.12_14.56.45.0.log
```



```
[9/12/21 14:56:45:652 GMT] 00000045 com.ibm.ws.logging.internal.impl.IncidentImpl           I FFDC1015I: An FFDC Incident has been  
created: "com.ibm.msg.client.jms.DetailedJMSEception: MQJCA1011: Failed to allocate a JMS connection.  
  
An internal error caused an attempt to allocate a connection to fail.  
  
See the linked exception for details of the failure. com.ibm.zosconnect.service.mq.OneWayMQServiceInvocation 0004" at  
ffdc_21.09.12_14.56.45.1.log
```



```
[9/12/21 14:56:45:652 GMT] 00000045 com.ibm.zosconnect.service.mq.MQServiceInvocation          E BAQM0056E: An unexpectedJMSEception  
occurred while processing a request for service 'mq.GetService'. The exception message was 'MQJCA1011: Failed to allocate a JMS  
connection.'.
```

Spacing added between lines to improve readability



The FFDC dump showing additional JMS information

```
-----Start of DE processing----- = [9/12/21 14:56:45:567 GMT]
Exception = com.ibm.mq.connector.DetailedResourceException
Source = com.ibm.ejs.j2c.poolmanager.FreePool.createManagedConnectionWithMCWrapper
probeid = 004
Stack Dump = com.ibm.mq.connector.DetailedResourceException: MQJCA1011: Failed to allocate a JMS connection., error code: MQJCA1011 An
internal error caused an attempt to allocate a connection to fail. See the linked exception for details of the failure.
at com.ibm.mq.connector.services.JCAExceptionBuilder.buildException(JCAExceptionBuilder.java:169)
at com.ibm.mq.connector.services.JCAExceptionBuilder.buildException(JCAExceptionBuilder.java:135)
at com.ibm.mq.connector.ConnectionBuilder.createConnection(ConnectionBuilder.java:162)
at com.ibm.mq.connector.outbound.ManagedConnectionFactoryImpl.createConnection(ManagedConnectionFactoryImpl.java:655)
at com.ibm.mq.connector.outbound.ManagedConnectionFactoryImpl.<init>(ManagedConnectionFactoryImpl.java:200)
at com.ibm.mq.connector.outbound.ManagedConnectionFactoryImpl.createManagedConnection(ManagedConnectionFactoryImpl.java:248)
at com.ibm.ejs.j2c.FreePool.createManagedConnectionWithMCWrapper(FreePool.java:1376)
at com.ibm.ejs.j2c.FreePool.createOrWaitForConnection(FreePool.java:1246)
at com.ibm.ejs.j2c.PoolManager.reserve(PoolManager.java:1438)
at com.ibm.ejs.j2c.ConnectionManager.allocateMCWrapper(ConnectionManager.java:574)
at com.ibm.ejs.j2c.ConnectionManager.allocateConnection(ConnectionManager.java:306)
at com.ibm.mq.connector.outbound.ConnectionFactoryImpl.createManagedJMSSession(ConnectionFactoryImpl.java:309)
at com.ibm.mq.connector.outbound.ConnectionFactoryImpl.createConnectionInternal(ConnectionFactoryImpl.java:252)
at com.ibm.mq.connector.outbound.ConnectionFactoryImpl.createConnection(ConnectionFactoryImpl.java:225)
...
at java.lang.Thread.run(Thread.java:818)
Caused by: com.ibm.msg.client.jms.DetailedJMSEException: JMSFMQ6312: An exception occurred in the Java(tm) MQI.
The Java(tm) MQI has thrown an exception describing the problem.
See the linked exception for further information.
at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
...
...
... 27 more
Caused by: com.ibm.mq.jmqi.JmqiException: CC=2;RC=2495;AMQ8568: The native JNI library 'mqjrrs64' was not found. For a client installation
this is expected. [3=mqjrrs64]
at com.ibm.mq.jmqi.local.LocalMQ.loadLib(LocalMQ.java:1178)
Caused by: java.lang.UnsatisfiedLinkError: /usr/lpp/mqm/V9R1M0/java/lib/libmqjrrs64.so (EDC5205S DLL module not found.)
```

Root cause – configuration issue in the MQ resource adapter configuration, e.g., nativeLibraryPath.

mitchj@us.ibm.com

© 2017, 2023 IBM Corporation
Slide 222



A FFDC dump showing an SSL Handshake issue

```
. . . -----Start of DE processing----- = [6/16/21 17:59:45:534 GMT]
Exception = java.security.cert.CertPathValidatorException
Source = com.ibm.ws.ssl.core.WSX509TrustManager
probeid = checkServerTrusted
Stack Dump = java.security.cert.CertPathValidatorException: The certificate issued by CN=OpenIdProv, OU=CertAuth is not trusted; internal cause is: java.security.cert.CertPathValidatorException: Certificate chaining error
at com.ibm.security.cert.BasicChecker.<init>(BasicChecker.java:111)
at com.ibm.security.cert.PKIXCertPathValidatorImpl.engineValidate(PKIXCertPathValidatorImpl.java:220)
at java.security.cert.CertPathValidator.validate(CertPathValidator.java:278)
at com.ibm.jsse2.util.f.a(f.java:40)
at com.ibm.jsse2.util.f.b(f.java:143)
. . .
e = class com.ibm.jsse2.util.f@5728f8dd
f = null
z = class java.lang.String[37]
tsCfgAlias = "OutboundKeyRing"
tsFile = "safkeyring:///zCEE.KeyRing"
extendedInfo = class java.util.HashMap@5ebd51b
serialVersionUID = 362498820763181265
```

Root cause – CA used to sign server certificate was not present in outbound key ring.

Tech-Tip: Use the Java JSSE debugging utility to enable SSL tracing at the Java level.

Use the Java runtime directive `-Djavax.net.debug` to enable this tracing by setting this directive value to `ssl`, e.g. **`-Djavax.net.debug=ssl`**. For more options regarding additional trace options SSL tracing available, see URL <https://www.ibm.com/docs/en/sdk-java-technology/8?topic=troubleshooting-debugging-utilities>

Using this directive requires the Java SDK be at Version 8, service release 6, fix pack 36 or later release level.



Tech/Tip: Use the Java directive javax.net.debug to enable Java SSL tracing

Add this directive to the JVM properties `-Djavax.net.debug=ssl,handshake`

```
.java:1168|JsseJCE: Using cipher DES/CBC/NoPadding from provider TBD via init
.java:1168|JsseJCE: Using cipher RC4 from provider TBD via init
.java:1168|JsseJCE: Using cipher DES/CBC/NoPadding from provider TBD via init
.java:1168|JsseJCE: Using cipher DESede/CBC/NoPadding from provider TBD via init
-
-
-
.java:1168|JsseJCE: Using cipher AES/GCM/NoPadding from provider TBD via init
.java:1168|JsseJCE: Using cipher ChaCha20-Poly1305 from provider TBD via init
-
-
-
.java:1168|JsseJCE: Using KeyGenerator IbmTlsExtendedMasterSecret from provider TBD via init
.java:1168|JsseJCE: Using signature SHA1withECDSA from provider TBD via init
.java:1168|JsseJCE: Using signature NONEwithECDSA from provider TBD via init
-
-
-
.java:1168|Consuming ClientHello handshake message (
-
-
-
.java:1168|Consumed extension: supported_versions
.java:1168|Negotiated protocol version: TLSv1.2
-
-
-
.java:1168|Produced ServerHello handshake message (
-
-
-
.java:1168|Produced server Certificate handshake message (
-
-
-
.java:1168|Produced ECDH ServerKeyExchange handshake message (
-
-
-
.java:1168|Produced ServerHelloDone handshake message (
-
-
-
.java:1168|Consuming ECDHE ClientKeyExchange handshake message (
-
-
-
.java:1168|Consuming ChangeCipherSpec message
-
-
-
.java:1168|Consuming client Finished handshake message (
-
-
-
.java:1168|Produced ChangeCipherSpec message
.java:1168|Produced server Finished handshake message (
-
-
-
```

For more details, see URL <https://www.ibm.com/docs/en/sdk-java-technology/8?topic=troubleshooting-debugging-utilities>



Other common TLS handshake issues

- ***Error occurred during a read, exception:javax.net.ssl.SSLHandshakeException: null cert chain***

This exception occurs when the server configuration set to require client certificates (`clientAuthentication="true"`) and the client had no certificate to provide and no alternative authentication method was available.

- ***Error occurred during a read, exception:javax.net.ssl.SSLEException: Received fatal alert: bad_certificate error (handshake), vc=1083934466
Caught exception during unwrap, javax.net.ssl.SSLEException: Received fatal alert: bad_certificate***

This is usually caused when the client certificate presented to the server did not have a certificate authority(CA) certificate for the CA that signed the client's personal certificate in the server's trust store key ring.

- ***CWWKO0801E: Unable to initialize SSL connection. Unauthorized access was denied or security settings have expired. Exception is javax.net.ssl.SSLHandshakeException: no cipher suites in common***

- There may be many causes for this issue but first confirm the RACF identity under which the server is running has either READ access to FACILITY resources IRR.DIGTCERT.LISTRING and IRR.DIGTCERT.LIST or access to RDATALIB resources if virtual keyrings are being used.

The first FACILITY resource gives the identity access to their own key ring and the second allows access to the certificates. Of if virtual keyrings are in use, then the identity needs READ or UPDATE authority to the `<ringOwner>.<ringName>.LST` resource in the RDATALIB class. READ access enables retrieving one's own private key, UPDATE access enables retrieving another's private key.

An alternative cause: For a TLS handshake to occur, the server must first have access to a private or site certificate that has a private key and the server must have access to that certificate's private key and no certificate with a private key is available.

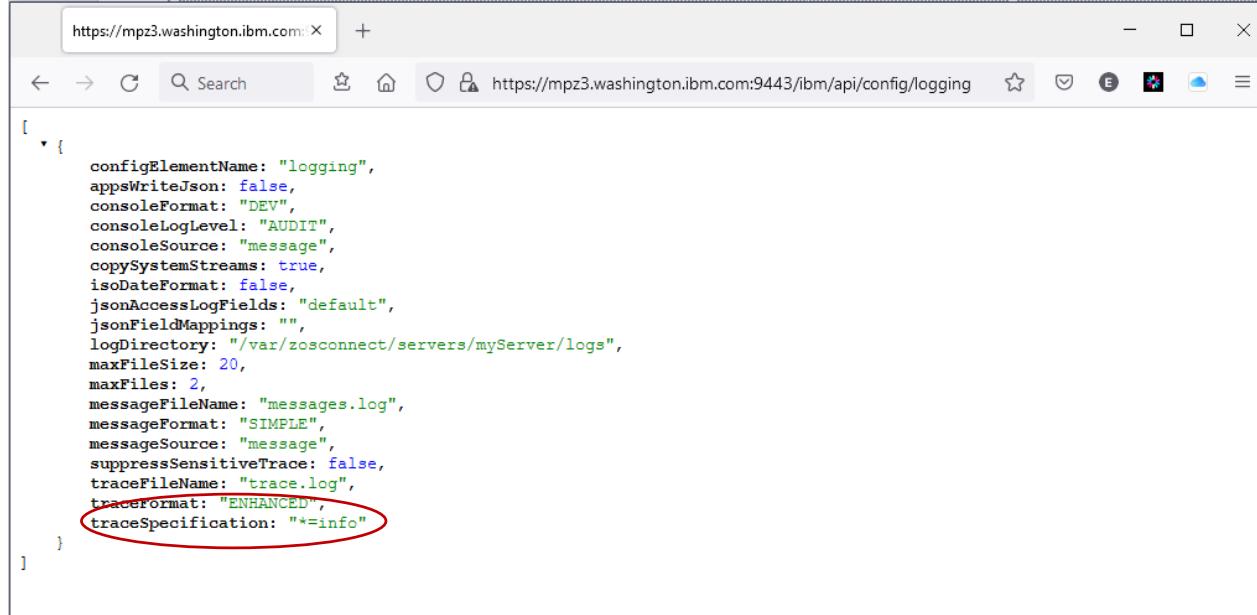
- Another possibility is that the TLS handshake the negotiations between the client and server failed, e.g., `javax.net.ssl.SSLHandshakeException: Client requested protocol SSLv3 is not enabled or supported in server context`



trace.out – use as a last resort or at the request of Level 2

First, the current active trace specification settings can be display using the *restConnector* feature.

`https://mpz3.washington.ibm.com:9443/ibm/api/config/logging`



```
[{"configElementName": "logging", "appsWriteJson": false, "consoleFormat": "DEV", "consoleLogLevel": "AUDIT", "consoleSource": "message", "copySystemStreams": true, "isoDateFormat": false, "jsonAccessLogFields": "default", "jsonFieldMappings": "", "logDirectory": "/var/zosconnect/servers/myServer/logs", "maxFileSize": 20, "maxFiles": 2, "messageFileName": "messages.log", "messageFormat": "SIMPLE", "messageSource": "message", "suppressSensitiveTrace": false, "traceFileName": "trace.log", "tracerFormat": "ENHANCED", "traceSpecification": "*=info"}]
```

Enabling trace in z/OS Connect EE server

<https://www.ibm.com/docs/en/zosconnect/3.0?topic=problems-enabling-trace-in-zos-connect-ee>



Managing trace specifications

- Use “include” file to save commonly used trace specifications.
- Add the “include” after the sever has started to avoid tracing the startup activity.

server.xml

```
<include location="${server.config.dir}/includes/safTrace.xml"/>
```

safTrace.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="security trace">
<logging traceSpecification="com.ibm.ws.security.*=all:
    SSLChannel=all:SSL=all:zosConnectSaf=all:zosConnect=all"/>
</server>
```

cicsTrace.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="CICS trace">
<logging traceSpecification="zosConnectServiceCics=all:
    com.ibm.zosconnect.wv*=FINEST:zosConnect=all"/>
</server>
```

imsTrace.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="IMS trace">
<logging traceSpecification="com.ibm.ims.*=all:
    com.ibm.j2ca.RAIMSTM=all:com.ibm.zosconnect.wv*=FINEST:
    zosConnect=all"/>
</server>
```

Enables enhanced tracing

(after adding an “include” file)
F BAQSTRT,REFRESH,CONFIG

Disable enhanced tracing

F BAQSTRT,LOGGING='*=INFO'

Or

F BAQSTRT,REFRESH,CONFIG
(after removing the “include” file)



trace.out file

mpz3

File Edit Settings View Communication Actions Window Help

File Edit Edit_Settings Menu Utilities Compilers Test Help

EDIT /MPZ3/usr/zosconnect/servers/myServer/logs/trace.log

Command ==>

003637 > getSSLConfig: DefaultSSLSettings Entry
003638 < getSSLConfig Exit
003639 SSLConfig.toString() {

003683 > determineIfCSIV2SettingsApply Entry
003684 (com.ibm.ssl.remoteHost:*, com.ibm.ssl.direction=inbound, com.ibm.ssl.remotePort=9443, com.ibm.ssl.endPointName=defaultHttpEndpoint-ssl)
003685 < determineIfCSIV2SettingsApply (original settings) Exit

003730 3 keyStoreType: JCERACFKS
003731 3 trustStoreType: JCERACFKS

003734 3 keyStore: safkeuring:///Liberty.KeyRing
003735 3 keyStoreName: CellDefaultKeyStore
003736 3 keyStorePassword: *****
003737 3 trustStore: safkeuring:///Liberty.KeyRing
003738 3 trustStoreName: CellDefaultKeyStore
003739 3 trustStorePassword: *****

003741 (com.ibm.ssl.remoteHost:*, com.ibm.ssl.direction=inbound, com.ibm.ssl.remotePort=9443, com.ibm.ssl.endPointName=defaultHttpEndpoint-ssl)
004117 K 3 Error occurred during a read, exception:javax.net.ssl.SSLHandshakeException: Empty server certificate chain
004119 3 Caught exception during unwrap, javax.net.ssl.SSLHandshakeException: Empty server certificate chain
004142 (com.ibm.ssl.remoteHost:*, com.ibm.ssl.direction=inbound, com.ibm.ssl.remotePort=9443, com.ibm.ssl.endPointName=defaultHttpEndpoint-ssl)
004144 > isTransportSecurityEnabled Entry
004145 < isTransportSecurityEnabled true Exit

004150 > getSSLConfig: DefaultSSLSettings Entry
004151 < getSSLConfig Exit
004152 SSLConfig.toString() {

004196 > determineIfCSIV2SettingsApply Entry
004197 (com.ibm.ssl.remoteHost:*, com.ibm.ssl.direction=inbound, com.ibm.ssl.remotePort=9443, com.ibm.ssl.endPointName=defaultHttpEndpoint-ssl)
004198 < determineIfCSIV2SettingsApply (original settings) Exit

004243 3 keyStoreType: JCERACFKS
004244 3 trustStoreType: JCERACFKS

004247 3 keyStore: safkeuring:///Liberty.KeyRing
004248 3 keyStoreName: CellDefaultKeyStore
004249 3 keyStorePassword: *****
004250 3 trustStore: safkeuring:///Liberty.KeyRing
004251 3 trustStoreName: CellDefaultKeyStore
004252 3 trustStorePassword: *****

004254 (com.ibm.ssl.remoteHost:*, com.ibm.ssl.direction=inbound, com.ibm.ssl.remotePort=9443, com.ibm.ssl.endPointName=defaultHttpEndpoint-ssl)
004630 K 3 Error occurred during a read, exception:javax.net.ssl.SSLHandshakeException: Empty server certificate chain
004632 3 Caught exception during unwrap, javax.net.ssl.SSLHandshakeException: Empty server certificate chain
004655 (com.ibm.ssl.remoteHost:*, com.ibm.ssl.direction=inbound, com.ibm.ssl.remotePort=9443, com.ibm.ssl.endPointName=defaultHttpEndpoint-ssl)
004657 > isTransportSecurityEnabled Entry
004658 < isTransportSecurityEnabled true Exit

Columns 00101 00252
Scroll ==> PAGE

- 4 Line(s) not Displayed
- 43 Line(s) not Displayed
- 44 Line(s) not Displayed
- 2 Line(s) not Displayed
- 1 Line(s) not Displayed
- 375 Line(s) not Displayed
- 1 Line(s) not Displayed
- 22 Line(s) not Displayed
- 1 Line(s) not Displayed
- 4 Line(s) not Displayed
- 43 Line(s) not Displayed
- 44 Line(s) not Displayed
- 2 Line(s) not Displayed
- 1 Line(s) not Displayed
- 375 Line(s) not Displayed
- 1 Line(s) not Displayed
- 22 Line(s) not Displayed
- 1 Line(s) not Displayed
- 1 Line(s) not Displayed
- 375 Line(s) not Displayed
- 1 Line(s) not Displayed

MAP A 03/019

Connected to remote server/host mpz3 using lu/pool MPZ30006 and port 23

Use thread number and/or package name to control which trace records are displayed

Monitoring Java, Liberty and z/OS Connect



Java Health Center – Monitors the Java environment

Configuring the Monitoring Agent using JVM directives

Java Directives

- Xhealthcenter:level=headless run without a client
- Dcom.ibm.java.diagnostics.healthcenter.headless.output.directory=/var/zcee/hcd directory where HCD will be stored
- Dcom.ibm.java.diagnostics.healthcenter.socket.readwrite=on collect socket sent/receive data
- Dcom.ibm.java.diagnostics.healthcenter.headless.files.to.keep=2 number of HCD files to retain
- Dcom.ibm.java.diagnostics.healthcenter.headless.delay.start=value=0 delay start value in minutes
- Dcom.ibm.java.diagnostics.healthcenter.headless.run.pause.duration=0 pause between runs, in minutes
- Dcom.ibm.java.diagnostics.healthcenter.headless.run.duration=0 run duration, in minutes
- Dcom.ibm.java.diagnostics.healthcenter.headless.run.number.of.runs=0 number of runs
- Dcom.ibm.diagnostics.healthcenter.readonly=on no client connections allowed

Add directives to bootstrap.properties or a JVM properties file, e.g.,

/var/zcee/properties/zceeHCD.properties

```
-Dcom.ibm.tools.attach.enable=yes  
-Xhealthcenter:level=headless -Dcom.ibm.java.diagnostics.healthcenter.headless.output.directory=/var/zcee/hcd  
    -Dcom.ibm.java.diagnostics.healthcenter.socket.readwrite=on -Dcom.ibm.diagnostics.healthcenter.readonly=on  
    -Dcom.ibm.java.diagnostics.healthcenter.headless.run.duration=5  
    -Dcom.ibm.java.diagnostics.healthcenter.headless.run.number.of.runs=1 #
```

All the health center directives should be on one line.

For details on these and other Health Center configuration properties, see URL

<https://www.ibm.com/docs/en/mon-diag-tools?topic=agent-health-center-configuration-properties>

Java Health Center – Monitoring Agent Configuration



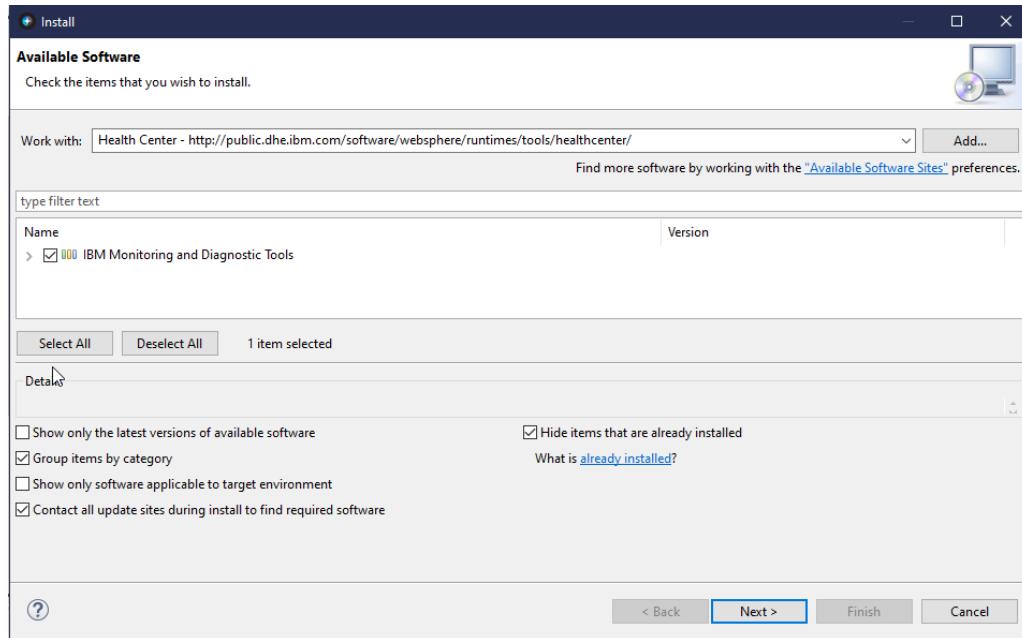
Set the JVM_OPTIONS environment variable to the properties file containing the health center directives

```
SYS1.PROCLIB(BAQSTRT)
//BAQSTRT PROC PARM='myServer --clean'
//*
// SET ZCONHOME='/usr/lpp/IBM/zosconnect/v3r0'
//*
//ZCON      EXEC PGM=BPXBATSL,REGION=0M,MEMLIMIT=8G,
//              PARM='PGM &ZCONHOME./bin/zosconnect run &PARMS.'
//STEPLIB   DD DISP=SHR,DSN=MQ91#.SCSQAUTH
//          DD DISP=SHR,DSN=MQ91#.SCSQANLE
//STDERR    DD SYSOUT=*,FREE=CLOSE,SPIN=(UNALLOC,1M)
//STDOUT    DD SYSOUT=*
//STDIN     DD DUMMY
//STDENV    DD *
_BPX_SHAREAS=YES
JAVA_HOME=/usr/lpp/java/J8.0_64/
WLP_USER_DIR=/var/zosconnect
JVM_OPTIONS=-Xoptionsfile=/var/zcee/properties/zceeHCD.properties
```

Java Health Center – Client Configuration



The Java health center client can be installed in most Eclipse workspace, e.g., IBM z/OS Explorer, etc.



The plug-in is available for download from <http://public.dhe.ibm.com/software/websphere/runtimes/tools/healthcenter/>

Java Health Center – HEAP analysis example



The screenshot shows the IBM Java Health Center interface within the Eclipse IDE. The main window displays a graph of Heap and pause times over time, showing used heap after collection, heap size, and pause time. Below the graph is a summary table of garbage collection metrics. To the right, a help panel provides information on using the garbage collection perspective.

Graph Legend:

- Used heap (after collection)
- Heap size
- Pause time

Summary Table Metrics:

Concurrent collection count	10
GC Mode	Default (gencon)
Global collections - Mean garbage collection pause	6.29 ms
Global collections - Mean interval between collections	2110 ms
Global collections - Number of collections	12
Largest memory request	199 KB
Mean garbage collection pause	3.5 ms
Mean interval between collections	129 ms
Minor collections - Mean garbage collection pause	3.39 ms
Minor collections - Mean interval between collections	134 ms
Minor collections - Number of collections	310
Minor collections - Total amount flipped	338073 KB
Minor collections - Total amount tenured	52.64 MB
Number of collections	322
Number of collections triggered by allocation failure	312
Proportion of time spent in garbage collection pauses (%)	2.71%
Proportion of time spent unpause (%)	97.29%
Rate of garbage collection	2643 MB/minute
Total amount flipped	338073 KB

Help Panel Content:

- Tool: IBM Monitoring and Diagnostic Tools - Health Center > IBM Monitoring and Diagnostic Tools - Health Center > Viewing the data collected > Garbage collection perspective
- Using the garbage collection perspective**
 - View data such as heap usage, pause times, summary table, object allocations, and tuning recommendation sections in the Health Center garbage collection perspective. Some data is not available for non-Java™ applications.
- The Health Center garbage collection perspective has the following views:
 - Views for basic garbage collection information
 - Views for detailed garbage collection information
- These views are available only for Java applications, and only if you enable detailed garbage collection information (Java applications only):
 - Object allocations: A table that shows the allocation of objects within a specified size range.
 - Samples by request site: A profile of sampled object allocations, grouped by the call site of the allocation request.
 - Samples by object: A profile of sampled object allocations, grouped by the type of object allocated.
 - Call hierarchy: This view shows data when you select a row in the Object allocations, Samples by request site, or Samples by object views. For example, if you select a row in the Samples by object view, this view shows the hierarchy of calls to allocations of that object.
 - Timeline: A visual indication of when object allocations were requested. This view shows data when you select a row in the Object allocations or Samples by request site views.

Java Health Center – Network analysis example



smf - Eclipse

File Edit Navigate Search Project Data Run Monitored System Window Help

Status Connection

CPU Classes Environment Events Garbage Collection I/O Locking Method Profiling Method Trace Native Memory Network Threads WebSphere Real Time

Analysis and Recommendations

- Your application has made 1,270 open socket requests and 820 close socket requests.
- Your application has 17 open sockets.
- No problems detected

Sockets

Socket ID filter:

ID	Type	IP Address	Port	Data sent	Data received	State	Thread [ID] Name
102	Client	0:0:0:ffff:c0a8:11c9	1491	116043 bytes	42284 bytes	Closed	[0x29d2fa00] Equino...
103	Client	0:0:0:ffff:c0a8:11c9	65470	32953 bytes	38334 bytes	Open	[0x2a00aa00] Default...
112	Server	0:0:0:ffff:c0a8:3c	59411			Open	[0x2a253d00] Shared...
127	Server	0:0:0:ffff:c0a8:3c	2446	87343 bytes	98768 bytes	Closed	[0x2a019f00] Default...
136	Server	0:0:0:ffff:c0a8:11c9	9080			Open	[0x2b38c800] Default...
138	ServerS...	0:0:0:0:0	59412	4248 bytes	8818 bytes	Open	[0x2a253d00] Shared...
144	Server	0:0:0:ffff:c0a8:3c	9443			Open	[0x2a019f00] Default...
164	ServerS...	0:0:0:0:0	176			Open	[0x2a253d00] Shared...
183	Client	0:0:0:ffff:c0a8:11c9	4000	182558 bytes	186691 bytes	Closed	[0x2a00aa00] Default...
186	Server	0:0:0:ffff:c0a8:11f3	7883			Open	[0x2a14f400] Default...
196	Server	0:0:0:ffff:c0a8:3c	61723			Closed	[0x29fcbb00] Default...
204	Server	0:0:0:ffff:c0a8:11f3	7880	1428 bytes	602 bytes	Open	[0x2a253d00] Shared...
215	Client	0:0:0:ffff:c0a8:11c9	1491	116825 bytes	62048 bytes	Open	[0x2b38c800] Default...
226	Server	0:0:0:ffff:c0a8:11f3	7863	2447 bytes	1059 bytes	Closed	[0x2a00aa00] Default...
227	Server	0:0:0:ffff:c0a8:11f3	9463	9892 bytes	8675 bytes	Open	[0x2aa3c100] Default...
228	Server	0:0:0:ffff:c0a8:11f3	7849			Closed	[0x29fcbb00] Default...
230	Server	0:0:0:ffff:c0a8:11f3	7850	39936 bytes	54048 bytes	Open	[0x2a00aa00] Default...
231	Server	0:0:0:ffff:c0a8:11f3	9463	10868 bytes	7460 bytes	Open	[0x2a14f400] Default...
233	Server	0:0:0:ffff:c0a8:11f3	7810	22059 bytes	11436 bytes	Open	[0x2a00aa00] Default...
234	Server	0:0:0:ffff:c0a8:11f3				Closed	[0x2a00aa00] Default...

Sockets open Network I/O

number (amount)

elapsed time (minutes)

c0a8:11c9 = 192.168.17.201

Java Health Center – Method Profiling



The screenshot shows the Java Health Center interface in Eclipse, specifically the Method Profiling section. The interface includes a navigation bar, toolbars, and several panes for monitoring system health and analyzing performance data.

Left Sidebar: Contains links for CPU, Classes, Environment, Events, Garbage Collection, I/O, Locking, Method Profiling (highlighted), Method Trace, Native Memory, Network, Threads, and WebSphere Real Time. It also includes Analysis and Recommendations, which states: "The method MD5.a() is consuming approximately 27% of the CPU cycles consumed by methods. It may be a good candidate for optimization." and "The monitored system generated more data than the client could consume, and so some samples have been lost. Profile accuracy should not be significantly affected."

Main Window: Features three main panes:

- Sample based profile:** A table showing method samples and their percentages. The top row highlights com.ibm.crypto.provider.MD5.a().
- Samples over time:** A graph showing the number of samples over time. A red circle highlights a peak around 2:30, and a blue arrow points to a second graph below.
- Invocation paths / Called methods / Timeline / Method trace summary:** Another table of samples and methods, identical to the first one.

Sample JCL - Restarting the Java Health Center collection

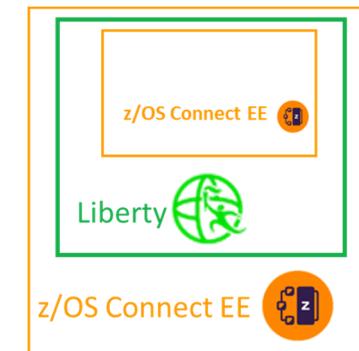


SDSF PROCESS DISPLAY MPZ3 ALL		LINE 1-5 (5)									
COMMAND INPUT ==> PS		SCROLL ==> CSR									
NP	JOBNAME	Status	Owner	State	CPU-Time	PID	PPID	ASID	ASIDX	LatchWaitPID	Command
BAQSTRT	WAITING FOR CHILD	LIBSERV	1W	40.01	69050	83955129	42	002A			/bin/sh /usr/lpp/IBM/zosconnect/v3r0/bin
BAQSTRT	OTHER KERNEL WAIT	LIBSERV	HK	40.01	16846267	69050	42	002A			/usr/lpp/java/J8.0_64/bin/java -javagen
BAQZANGL	SWAPPED, RUNNING	LIBANGE	1RI	0.01	50399398	83953829	77	004D			/usr/lpp/IBM/zosconnect/v3r0/wlplib/nat
BAQZANGL	SWAPPED, FILE SYS KERNEL WAIT	LIBANGE	1FI	0.01	83953829		1	77	004D		BPXBATA2
BAQSTRT	FILE SYS KERNEL WAIT	LIBSERV	1F	40.01	83955129		1	42	002A		BPXBATSL

```
*****
product = WAS FOR z/OS 21.0.0.9, z/OS Connect 03.00.52 (wlp-1.0.56.cl210920210909-1618)
wlp.install.dir = /shared/IBM/zosconnect/v3r0/wlp/
server.config.dir = /var/zosconnect/servers/myServer/
java.home = /shared/java/J8.0_64
java.version = 1.8.0_301
java.runtime = Java(TM) SE Runtime Environment (8.0.6.36 - pmz6480sr6fp36-20210913_01(SR6 FP36))
os = z/OS (02.03.00; s390x) (en_US)
process = 16846267@wg31
*****
```

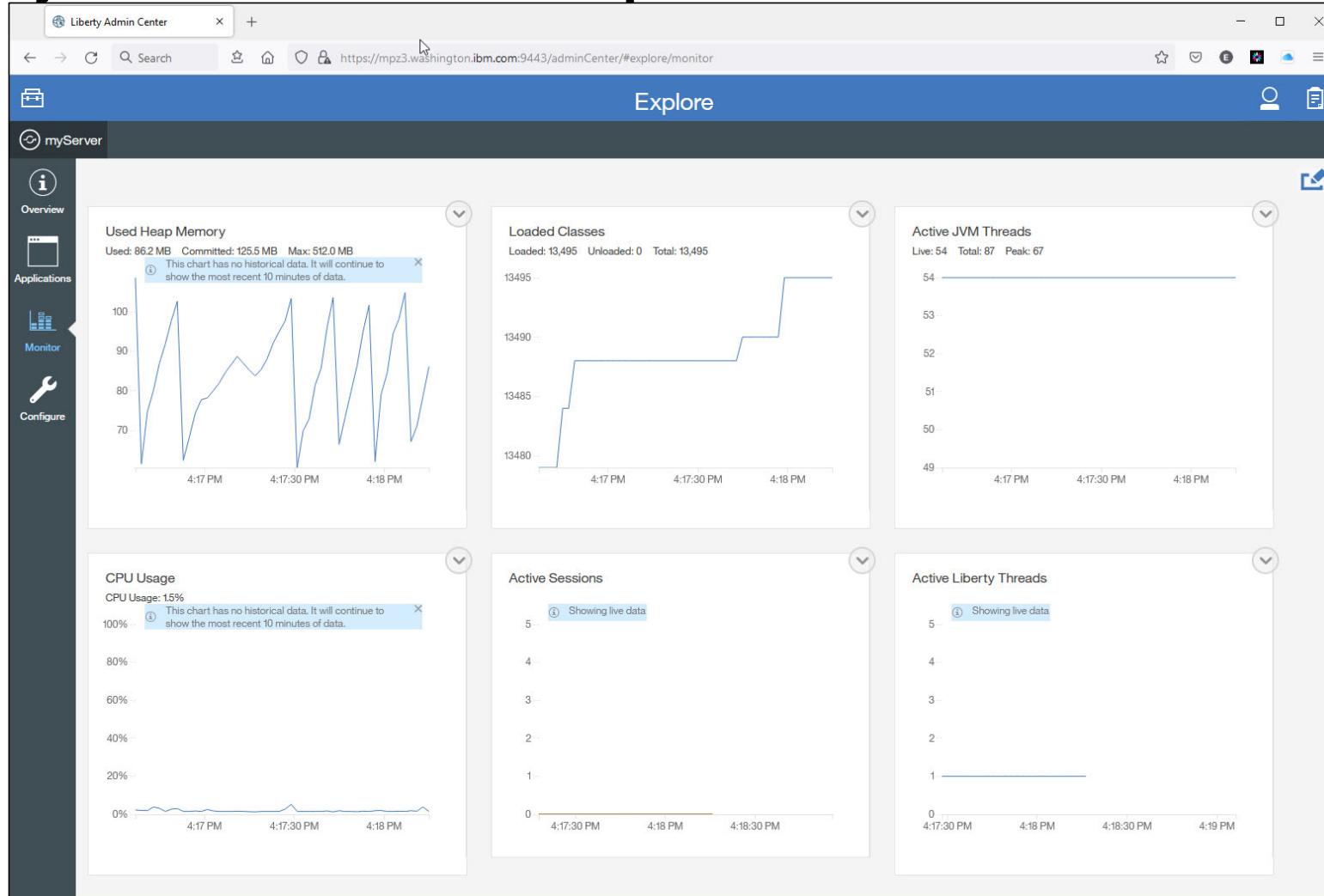
```
//JOHNSONS JOB (ACCOUNT), NOTIFY=&SYSUID,REGION=0M,
// CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),USER=LIBSERV
//JAVA      EXEC PGM=IKJEFT01,REGION=0M
//SYSERR   DD   SYOUT=*
//STDOUT    DD   SYOUT=*
//SYSTSPRT DD   SYOUT=*
//SYSTSIN  DD   *
BPXBATCH SH +
java -jar /usr/lpp/java/J8.0_64/lib/ext/healthcenter.jar +
ID=16846267 level=headless +
-Dcom.ibm.java.diagnostics.healthcenter.headless.run.number.of.runs=1
```

The job must be executed under the same identity under which the server is running.





Liberty Admin Center feature provides real time monitoring



Workload Manager - Definitions

WLM Report Classes

Report Class Selection List Row 1 to 12 of 12

Action Codes: 1=Create, 2=Copy, 3=Modify, 4=Browse, 5=Print, 6=Delete, /=Menu Bar

Action	Name	Description	User	Date
BAOSTC			JOHNSON	2021/09/04
WMQFTE			JOHNSON	2021/08/31
WMQFTER			JOHNSON	2021/08/31
WMQFTEZ			JOHNSON	2021/08/31
ZCEEADM			JOHNSON	2021/08/02
ZCEEAPIR			JOHNSON	2021/08/05
ZEECICS			JOHNSON	2021/08/05
ZCEEDB2			JOHNSON	2021/08/05
ZEEIMS			JOHNSON	2021/08/05
ZCEEMQ			JOHNSON	2021/08/05
ZCEEOTHR			JOHNSON	2021/08/02
ZCEESTC			JOHNSON	2021/09/02

***** Bottom of data *****

Service-Class Xref Notes Options Help

Modify a Service Class Row 1 to 2 of 2

Command ==>

Service Class Name : OPS_HIGH
 Description : System Tasks Velocity 70
 Workload Name : STC_WKL (name or ?)
 Base Resource Group : (name or ?)
 Cpu Critical : NO (YES or NO)
 I/O Priority Group : NORMAL (NORMAL or HIGH)
 Honor Priority : DEFAULT (DEFAULT or NO)

Specify BASE GOAL information. Action Codes: I=Insert new period, E>Edit period, D=Delete period.

Action	#	Duration	Imp.	Description
	1	1		Execution velocity of 70

***** Bottom of data *****

mitchj@us.ibm.com 19/004
 Connected to remote server/host mpz3 using lu/pool MPZ30008 and port 23

WLM "CB" Classification Rules

SubSystem-Type Xref Notes Options Help

Modify Rules for the Subsystem Type Row 1 to 8 of 16

Command ==>

Subsystem Type . . . : CB Fold qualifier names? N Y or N
 Description : WLP/zCEE Transactions

Action codes: A=After C=Copy M=Move I=Insert rule
 B=Before D=Delete row R=Repeat IS=Insert Sub-rule
 More ==>

Action	Type	Name	Start	Service	Report
1	CN	myServer		OPS_HIGH	ZCEEOTHR
2	TC	TCAPIR		OPS_HIGH	BAOSTC
2	TC	TCCICS		OPS_HIGH	ZCEEAPIR
2	TC	TCDB2		OPS_HIGH	ZCEEDB2
2	TC	TCIMS		OPS_HILO	ZEEIMS
2	TC	TCMQ		OPS_MED	ZCEEMQ
2	TC	TCOTHR		OPS_LOW	ZCEEOTHR

-----Class-----

Defaults: OPS_HIGH ZCEEOTHR
 OPS_HIGH ZCEESTC
 OPS_HIGH ZCEEADM
 OPS_HIGH ZCEEAPIR
 OPS_HIGH ZEECICS
 OPS_HIGH ZCEEDB2
 OPS_HILO ZEEIMS
 OPS_MED ZCEEMQ
 OPS_LOW ZCEEOTHR

07/021

SubSystem-Type Xref Notes Options Help

Modify Rules for the Subsystem Type Row 9 to 16 of 16

Command ==>

Subsystem Type . . . : CB Fold qualifier names? N Y or N
 Description : WLP/zCEE Transactions

Action codes: A=After C=Copy M=Move I=Insert rule
 B=Before D=Delete row R=Repeat IS=Insert Sub-rule
 More ==>

Action	Type	Name	Start	Service	Report
1	CN	zceex*		OPS_HIGH	ZCEEOTHR
2	TC	TCAPIR		OPS_HIGH	ZCEESTC
2	TC	TCCAPIR		OPS_HIGH	ZCEEADM
2	TC	TCCICS		OPS_HIGH	ZCEEAPIR
2	TC	TCDB2		OPS_HIGH	ZCEEDB2
2	TC	TCIMS		OPS_HILO	ZEECICS
2	TC	TCMQ		OPS_HILO	ZEEIMS
2	TC	TCOTHR		OPS_MED	ZCEEMQ

-----Class-----

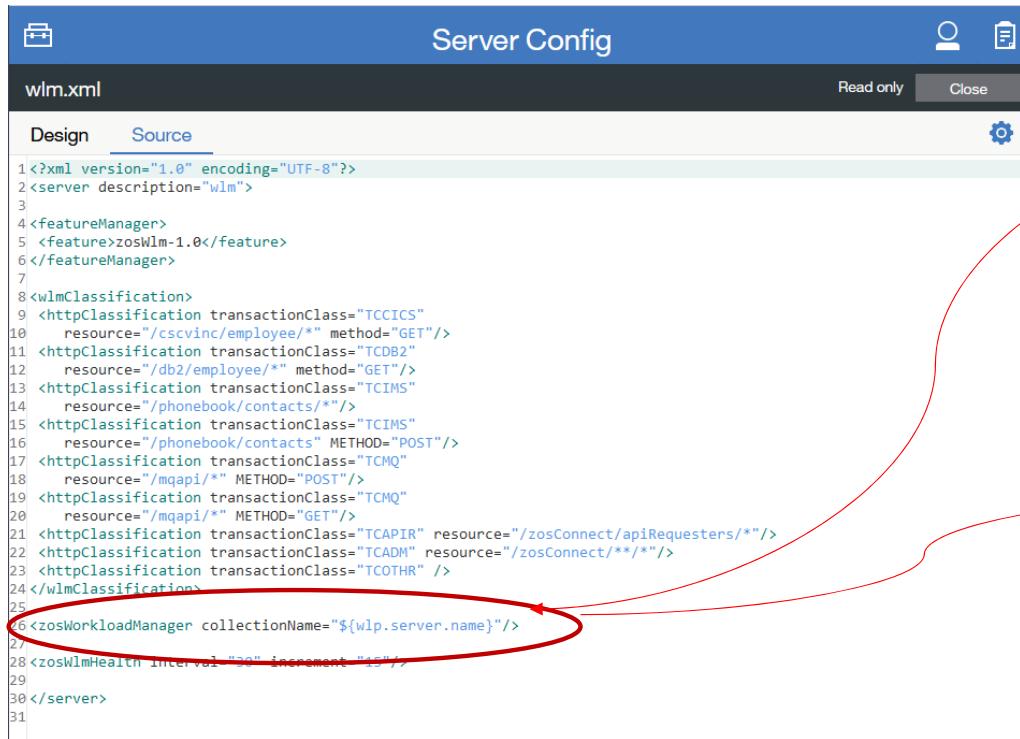
Defaults: OPS_HIGH ZCEEOTHR
 OPS_HIGH ZCEESTC
 OPS_HIGH ZCEEADM
 OPS_HIGH ZCEEAPIR
 OPS_HIGH ZEECICS
 OPS_HILO ZEEIMS
 OPS_MED ZCEEMQ
 OPS_HILO ZCEEOTHR

07/021

Workload Manager – WLM Classification server XML

The corresponding required server XML configuration

- Based on HTTP path matching (port and/or method can also be specified)
- The default value for the *wlmClassification* name is the name of the server
- See URL <https://www.ibm.com/docs/en/was-liberty/zos?topic=zos-wlm-classification> for more information
- The *transactionClass* attribute is required to ensure an enclave is created.



Server Config

wlm.xml

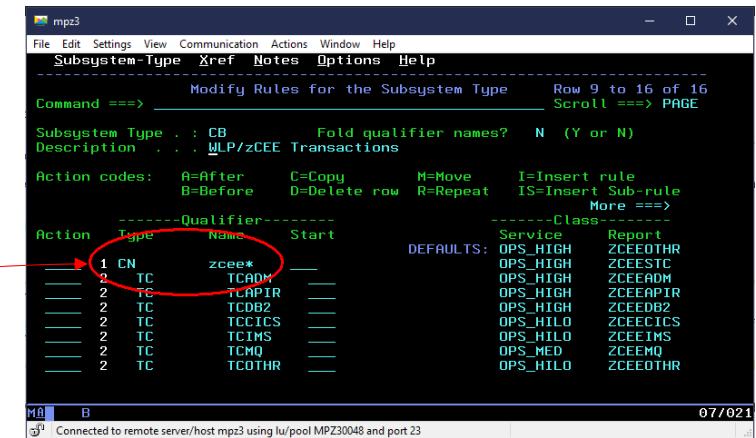
Design Source

```

1<?xml version="1.0" encoding="UTF-8"?>
2<server description="wlm">
3
4<featureManager>
5 <feature>zosWlm-1.0</feature>
6</featureManager>
7
8<wlmClassification>
9 <httpClassification transactionClass="TCCICS"
10   resource="/cscvinc/employee/*" method="GET"/>
11 <httpClassification transactionClass="TCDB2"
12   resource="/db2/employee/*" method="GET"/>
13 <httpClassification transactionClass="TCIMS"
14   resource="/phonebook/contacts/*"/>
15 <httpClassification transactionClass="TCIMS"
16   resource="/phonebook/contacts" METHOD="POST"/>
17 <httpClassification transactionClass="TCMQ"
18   resource="/mqapi/*" METHOD="POST"/>
19 <httpClassification transactionClass="TCMQ"
20   resource="/mqapi/*" METHOD="GET"/>
21 <httpClassification transactionClass="TCAPIR" resource="/zosConnect/apiRequesters/*"/>
22 <httpClassification transactionClass="TCADM" resource="/zosConnect/**/*"/>
23 <httpClassification transactionClass="TCOTHR" />
24</wlmClassification>
25
26<zosWorkloadManager collectionName="${wlp.server.name}">
27
28<zosWlmHealth interval="10" increment="15"/>
29
30</server>
31

```

Related to WLM CN name.



mpz3

File Edit Settings View Communication Actions Window Help

Subsystem-Type . : CB Fold qualifier names? N (Y or N)

Description . . . WLP/zCEE Transactions

Action codes: A=After C=Copy M=Move I=Insert rule
B=Before D=Delete row R=Repeat IS=Insert Sub-rule
More ==>

Action	Type	Name	Start	Service	Report
1	CN	zcee*		OPSLHIGH	ZCEE0THR
2	TC	TCADM		OPSLHIGH	ZCEE0STC
2	TC	TCDB2		OPSLHIGH	ZCEEADM
2	TC	TCCICS		OPSLHIGH	ZCEEAPTR
2	TC	TCIMS		OPSLHILO	ZCEEICCS
2	TC	TCMQ		OPSLHILO	ZCEEIMMS
2	TC	TCOTHR		OPSMED	ZCEE0MQ
				OPSHILO	ZCEE0THR

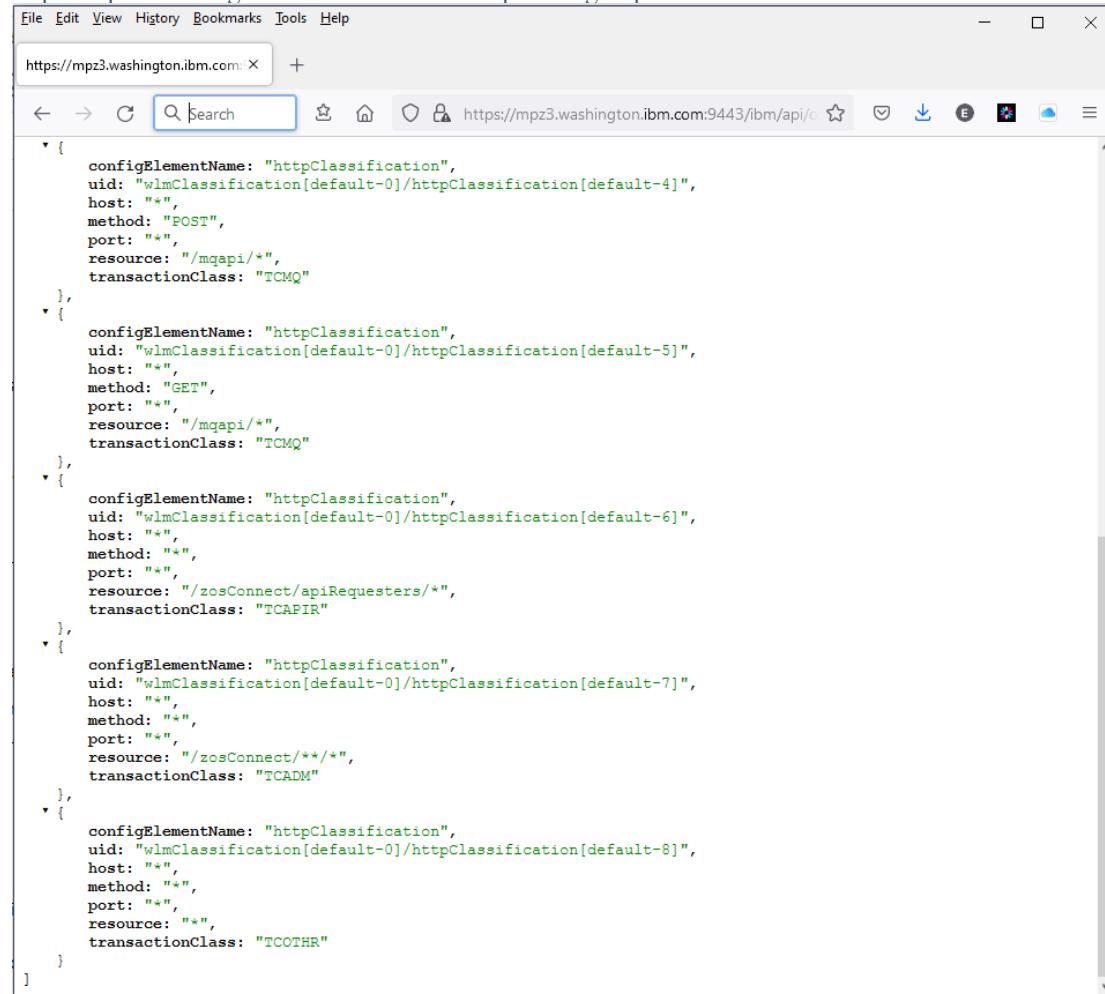
Defaults: OPS_HIGH ZCEE0THR
OPS_HIGH ZCEE0STC
OPS_HIGH ZCEEADM
OPS_HIGH ZCEEAPTR
OPS_HIGH ZCEEICCS
OPSLHILO ZCEEIMMS
OPSMED ZCEE0MQ
OPSHILO ZCEE0THR

MA B 07/021
Connected to remote server/host mpz3 using lu/pool MPZ30048 and port 23



Workload Manager – Active HTTP Classification

<https://mpz3.washington.ibm.com:9443/ibm/api/config/httpClassification>



The screenshot shows a web browser window displaying a JSON array of configuration elements for Workload Manager Active HTTP Classification. The URL in the address bar is <https://mpz3.washington.ibm.com:9443/ibm/api/config/httpClassification>. The JSON data lists nine configuration elements, each defined by its configElementName, uid, host, method, port, resource, and transactionClass. The transactionClasses listed are TCMQ, TCMQ, TCAPIR, TCADM, and TCOTHR.

```
[{"configElementName": "httpClassification", "uid": "wlmClassification[default-0]/httpClassification[default-4]", "host": "*", "method": "POST", "port": "*", "resource": "/mqapi/*", "transactionClass": "TCMQ"}, {"configElementName": "httpClassification", "uid": "wlmClassification[default-0]/httpClassification[default-5]", "host": "*", "method": "GET", "port": "*", "resource": "/mqapi/*", "transactionClass": "TCMQ"}, {"configElementName": "httpClassification", "uid": "wlmClassification[default-0]/httpClassification[default-6]", "host": "*", "method": "*", "port": "*", "resource": "/zosConnect/apiRequesters/*", "transactionClass": "TCAPIR"}, {"configElementName": "httpClassification", "uid": "wlmClassification[default-0]/httpClassification[default-7]", "host": "*", "method": "*", "port": "*", "resource": "/zosConnect/**/*", "transactionClass": "TCADM"}, {"configElementName": "httpClassification", "uid": "wlmClassification[default-0]/httpClassification[default-8]", "host": "*", "method": "*", "port": "*", "resource": "*", "transactionClass": "TCOTHR"}]
```

RMF SMF Type 72 Service Class Reports

mpz3

File Edit Settings View Communication Actions Window Help

Display Filter View Print Options Search Help

SDSF OUTPUT DISPLAY JOHNSONR JOB12740 DSID 112 LINE CHARS 'CICS' FOUND

COMMAND INPUT ==>

POLICY=WSCPOL REPORT CLAS

-TRANSACTIONS--		TRANS-TIME	HHH.MM.SS.FFFFFF	TRA
AVG	0.02	ACTUAL	108891	TOT
MPL	0.02	EXECUTION	108856	MOB
ENDED	96	QUEUED	34	CAT
END/S	0.16	R/S AFFIN	0	CAT
#SWAPS	0	INELIGIBLE	0	
EXCTD	0	CONVERSION	0	
		STD DEV	762583	

---SERVICE----		SERVICE TIME	--APPL %---	--P	
IOC	0	CPU	1.967	CP 0.02	BLK
CPU	1739K	SRB	0.000	IIPCP 0.02	ENQ
MSO	0	RCT	0.000	IIP 0.31	CRM
SRB	0	IIT	0.000	AAPCP 0.00	LCK
TOT	1739K	HST	0.000	AAP N/A	SUP
/SEC	2898	IIP	1.844		
ABSRPTN	166K	AAP	N/A		
TRX SERV	166K				

MA A

Connected to remote server/host mpz3 using lu/pool MPZ30008 and port 23

mpz3

File Edit Settings View Communication Actions Window Help

Display Filter View Print Options Search Help

SDSF OUTPUT DISPLAY JOHNSONR JOB12740 DSID 112 LINE CHARS 'APIR' FOUND

COMMAND INPUT ==>

POLICY=WSCPOL REPORT CLASS=ZCEEAPIR PERIOD=1

-TRANSACTIONS--		TRANS-TIME	HHH.MM.SS.FFFFFF	TRANS-APPL%----CP-IIPCP/AAPCP-IIP/AAP	---ENCLAVES---	
AVG	0.14	ACTUAL	424835	TOTAL 0.12	0.12 0.73	Avg ENC 0.14
MPL	0.14	EXECUTION	424707	MOBILE 0.00	0.00 0.00	REM ENC 0.00
ENDED	200	QUEUED	126	CATEGORYA 0.00	0.00 0.00	MS ENC 0.00
END/S	0.33	R/S AFFIN	0	CATEGORYB 0.00	0.00 0.00	
#SWAPS	0	INELIGIBLE	0			
EXCTD	0	CONVERSION	0			
		STD DEV	1.381943			

---SERVICE----		SERVICE TIME	--APPL %---	--PROMOTED--	--DASD I/O--	---STORAGE----	-PAGE-IN RATES-
IOC	0	CPU	5.073	CP 0.12	BLK 0.000	SSCHRT 2.4	Avg 0.00 SINGLE 0.0
CPU	4485K	SRB	0.000	IIPCP 0.12	ENQ 0.000	RESP 0.4	TOTAL 0.00 BLOCK 0.0
MSO	0	RCT	0.000	IIP 0.73	CRM 0.000	CONN 0.3	SHARED 0.00 SHARED 0.0
SRB	0	IIT	0.000	AAPCP 0.00	LCK 0.000	DISC 0.0	
TOT	4485K	HST	0.000	AAP N/A	SUP 0.000	Q+PEND 0.0	
/SEC	7474	IIP	4.363			IOSQ 0.0	
ABSRPTN	53K	AAP	N/A				
TRX SERV	53K						

MA A

05/057

Connected to remote server/host mpz3 using lu/pool MPZ30008 and port 23



Liberty SMF 120 Subtype 11

WebSphere Liberty Profile (WLP) can generate various types of SMF 120 records. Support for a SMF 120 record relevant for z/OS Connect was added in WLP V16.0.0.2. This record, a SMF 120 Subtype 11, is generated for each HTTP request received by the Liberty server. For more details and a description of the contents of this record, see URL <https://www.ibm.com/support/pages/liberty-zos-smf-120-11-version-2>



The screenshot shows the 'Server Config' interface with a blue header bar. In the center, it says 'Server Config'. On the left is a small icon of a server. On the right are icons for search, refresh, and close. Below the header, the title 'smf.xml' is displayed, followed by 'Read only' and a 'Close' button. There are two tabs: 'Design' and 'Source'. The 'Source' tab is selected, showing the XML code for the SMF configuration. The code is as follows:

```
1<?xml version="1.0" encoding="UTF-8"?>
2
3<server description="SMF">
4    <featureManager>
5        <feature>monitor-1.0</feature>
6        <feature>zosRequestLogging-1.0</feature>
7    </featureManager>
8
9</server>
10
```

Useful Plug-ins for WAS z/OS SMF 120.9 Browser

<https://www.ibm.com/support/pages/node/6355403>

Liberty SMF 120 Subtype 11 – WP102312 Plugin



LibertyExport.csv

Mitch Johnson M

File Home Insert Page Layout Formulas Data Review View Help ACROBAT

Cut Copy Format Painter

Font Alignment Number Styles

Clipboard

General Conditional Formatting Table Normal Bad Good Neutral Calculation Check Cell

Insert Delete Format Cells

AutoSum Fill Sort & Filter Ideas Sensitivity

Share Comments

AS9 : 166

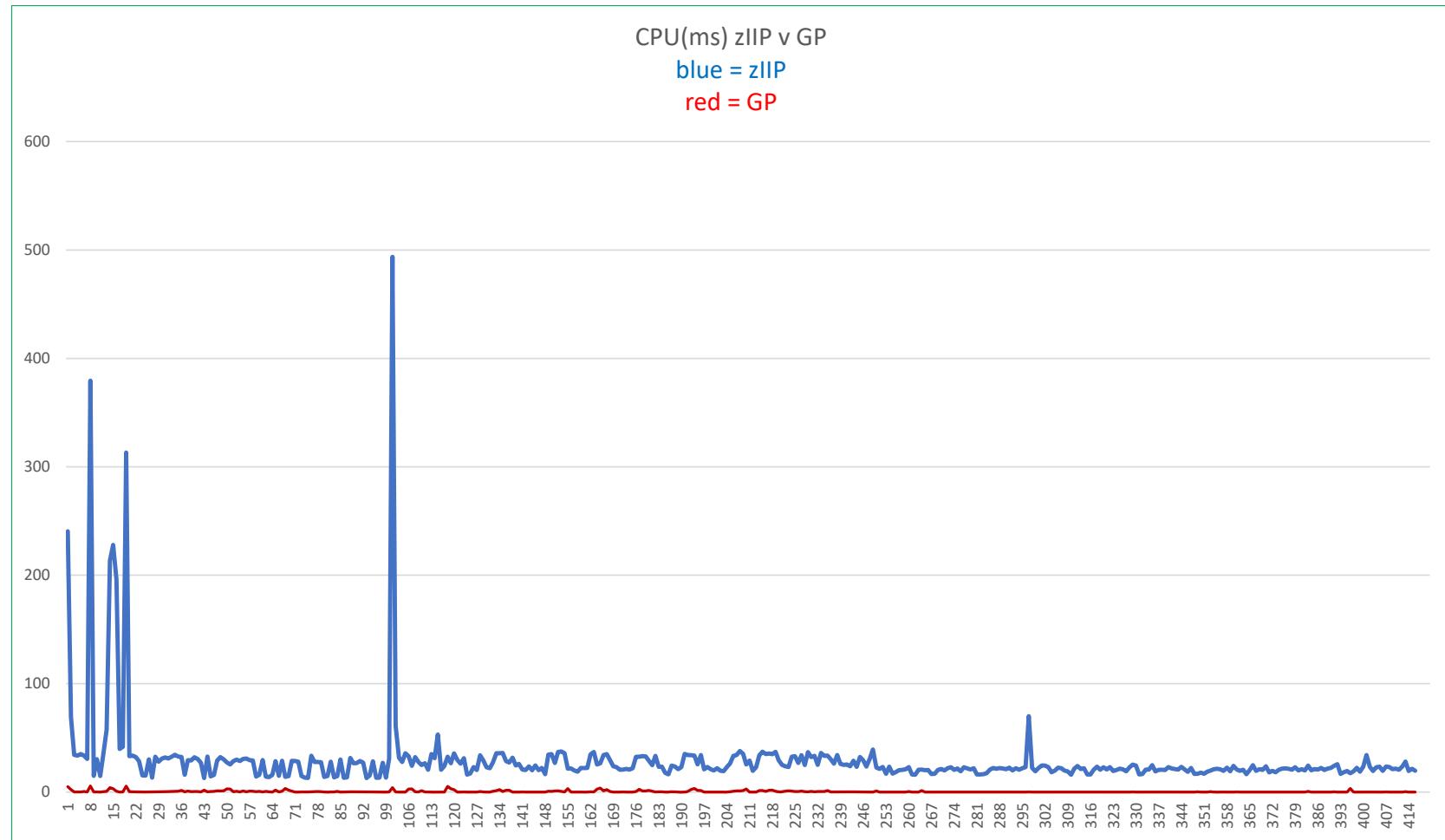
AS AT AU AV AW

1	SystemName	Sysplex	JobName	StartTime	StartTime	EndTime	EndTime	Respon	TranClass	Total CPU	Start Total CPU	E Total CPU	Total IGP(ms)	TotalOffload(ms)	userid	mappedU	requestUF	host port	uri	response	targetPort	remotePo	RemoteAddr
2	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	6080	TCAPIR	3314772936	4.52E+09	245.5195	5.0110927	240.50838	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4283	192.168.17.243	
3	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	7030	TCAPIR	178821759	471705165	71.51572	2.334169	69.18156	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4286	192.168.17.243	
4	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	374	TCAPIR	4327455460	4.469E+09	34.44008	0.10757129	34.332504	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4301	192.168.17.243	
5	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	495	TCAPIR	2762287407	2.9E+09	33.65053	0.057430662	33.5931	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4304	192.168.17.243	
6	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	500	TCAPIR	4484655211	4.629E+09	35.15451	0.12540185	35.020004	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4303	192.168.17.243	
7	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	262	TCAPIR	4637789017	4.777E+09	34.10283	0.42818993	33.680042	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4305	192.168.17.243	
8	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	293	TCAPIR	542458283	668050357	30.66213	0.053870115	30.608257	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4306	192.168.17.243	
9	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	10493	TCAPIR	3802597962	5.38E+09	385.0374	5.576215	379.46115	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4285	192.168.17.243	
10	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	185	TCAPIR	5384541333	5.446E+09	15.04486	0.15656103	14.888303	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4308	192.168.17.243	
11	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	282	TCAPIR	1028119195	1.153E+09	30.38298	0.04661279	30.336363	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4309	192.168.17.243	
12	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	163	TCAPIR	901260513	962209631	14.88016	0	14.880165	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4310	192.168.17.243	
13	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	5126	TCAPIR	3137255105	3.284E+09	35.92899	0.33009765	35.598892	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4313	192.168.17.243	
14	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	5122	TCAPIR	4890213483	5.128E+09	58.01673	0.61064285	57.40609	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4314	192.168.17.243	
15	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	24315	TCAPIR	13036032356	1.393E+10	217.4406	4.0119	213.4287	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4282	192.168.17.243	
16	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	3438	TCAPIR	1463812131	2.41E+09	290.9845	3.1036336	277.8809	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4284	192.168.17.243	
17	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	12587	TCAPIR	1160912461	1.967E+09	196.8579	0.7669092	196.09096	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4315	192.168.17.243	
18	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	6599	TCAPIR	5303866625	5.467E+09	39.79177	0.020269532	39.761494	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4316	192.168.17.243	
19	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	6565	TCAPIR	6143860672	6.315E+09	41.86705	0.16208105	41.704967	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4317	192.168.17.243	
20	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	25052	TCAPIR	2622790027	3.928E+09	318.7149	5.498493	313.22546	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4281	192.168.17.243	
21	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	7709	TCAPIR	4477460136	4.615E+09	33.52233	0.35891944	33.163406	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4322	192.168.17.243	
22	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	7682	TCAPIR	1973032107	2.112E+09	33.81701	0.19548193	33.621525	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4321	192.168.17.243	
23	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	14950	TCAPIR	458083508	590213570	32.25832	0.0489917	32.209324	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4323	192.168.17.243	
24	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	14016	TCAPIR	61401222	178390269	28.56178	0.2347461	28.327032	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4325	192.168.17.243	
25	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	14088	TCAPIR	86069826	148846164	15.32625	0.0541626	15.272091	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4326	192.168.17.243	
26	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	14097	TCAPIR	5471350509	5.535E+09	15.43587	0.21740967	15.218459	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4324	192.168.17.243	
27	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	7051	TCAPIR	5358173556	5.482E+09	30.16547	0.001757324	30.163715	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4328	192.168.17.243	
28	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	7029	TCAPIR	2281578411	2.336E+09	13.27289	0	13.272889	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4327	192.168.17.243	
29	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	379	TCAPIR	1054429318	1.188E+09	32.66632	0.067269534	32.599052	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4329	192.168.17.243	
30	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	347	TCAPIR	644045567	759168227	28.10612	0.16462207	27.941496	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4330	192.168.17.243	
31	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	18550	TCAPIR	764059849	891747729	31.1738	0.4028291	30.770971	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4336	192.168.17.243	
32	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	18551	TCAPIR	5678912186	5.811E+09	32.35731	0.39294335	31.964365	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4332	192.168.17.243	
33	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	18557	TCAPIR	260836676	390012335	31.53703	0.6369346	30.900091	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4331	192.168.17.243	
34	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	18568	TCAPIR	252264630	387487083	33.01329	0.4126411	32.600655	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4333	192.168.17.243	
35	MPZ3	MPZPLEX	BAQSTRT	Friday Au	3.84E+12	Friday Au	3.84E+12	18571	TCAPIR	6167008451	6.311E+09	35.09796	0.69125974	34.406696	USER1	/zosConn/mpz3.was	9080	/zosConnect/apiRequeste	166	9080	4334	192.168.17.243	

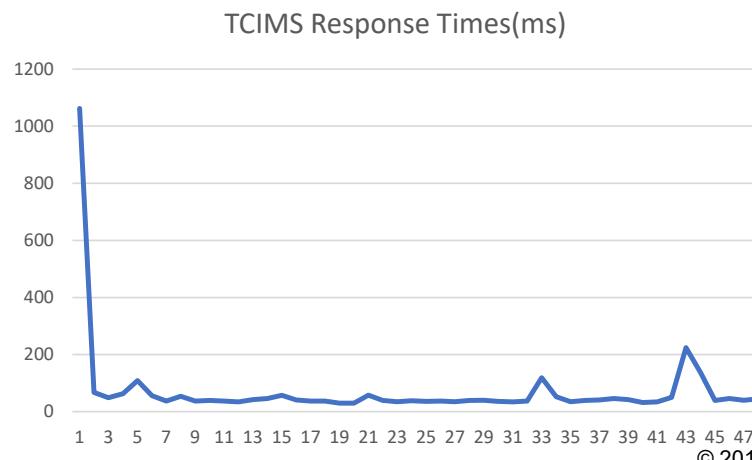
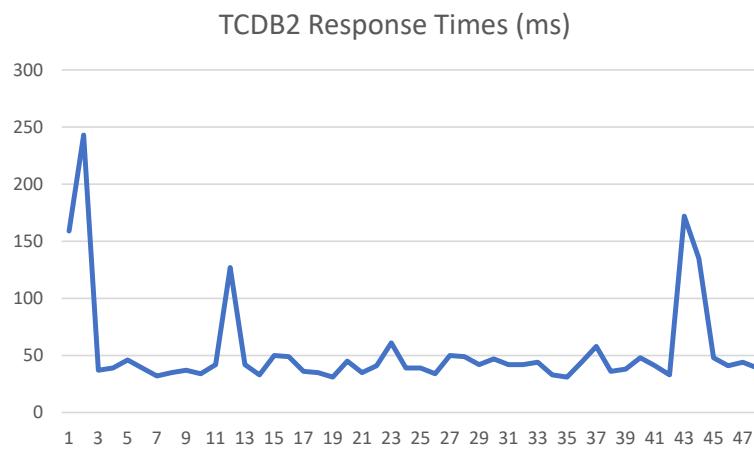
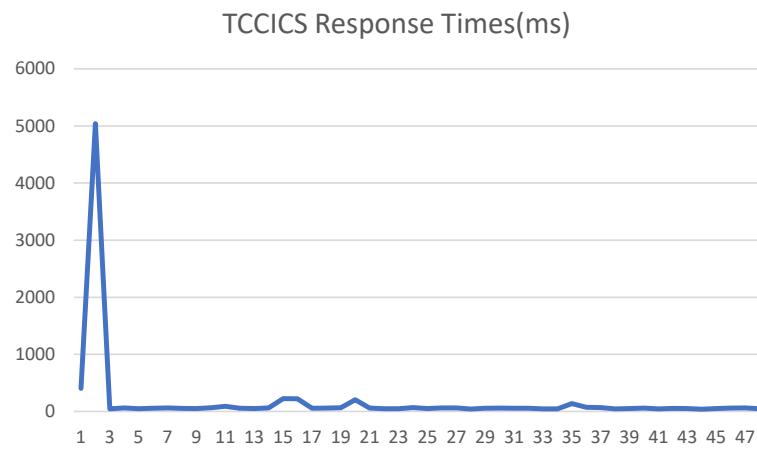
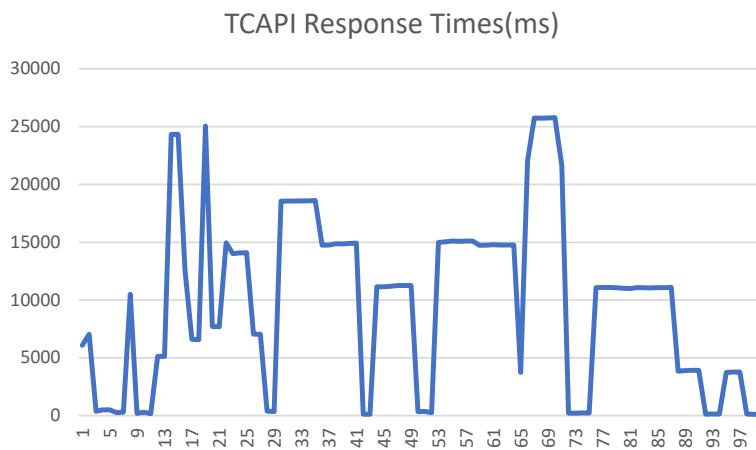
Some fields have been hidden



Liberty SMF 120 type 11 – GP v zIIP comparison example



Liberty SMF 120 type 11 – Response times comparisons example



z/OS Connect SMF 123 server XML configuration (OpenAPI 2)



SMF 123 records have two subtypes, and each subtype can have different versions.

- SMF type 123 subtype 1 records - Version 1 contains some basic information about both API provider and API requester requests. Version 2 supersedes version 1 and contains more detailed information about each API provider request, including information about to which system of record (SOR) the request was sent
- *SMF type 123 subtype 2 records - Version 2 supersedes subtype 1 version 1 and contains more detailed information about each API requester request, including information about to what HTTP endpoint the request was sent.*

Server Config

audit.xml

Read only Close

Design Source

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="SMF reporting">
  <zosconnect_zosConnectManager>
    globalInterceptorsRef="interceptorList_g"/>
  <zosconnect_authorizationInterceptor id="auth">
    safCacheTimeout="600"/>
  <zosconnect_auditInterceptor id="audit">
    apiRequesterSmfVersion="2"
    apiProviderSmfVersion="2"/>
  <zosconnect_zosConnectInterceptors id="interceptorList_g">
    interceptorRef="audit"/>
</server>
```

Server Config

audit.xml

Read only Close

Design Source

Server

z/OS Connect Manager

z/OS Connect Authorization Interceptor auth

z/OS Connect EE SMF Audit Interceptor audit

z/OS Connect Interceptors interceptorList_g

Sequence
0 (default)

The sequence in which this interceptor should be processed with respect to other configured interceptors implementing z/OS Connect's com.ibm.wsspi.zos.connect.Interceptor Service Provider Interface (SPI).

API provider SMF Version
2

The version of SMF 123 subtype 1 records to be written.

auditApiProviderRequestHeaders.name
(no value)

auditApiProviderRequestHeaders.desc

auditApiProviderResponseHeaders.name
(no value)

auditApiProviderResponseHeaders.desc

API requester SMF Version
2

The version of SMF 123 subtype 1 or subtype 2 records to be written.

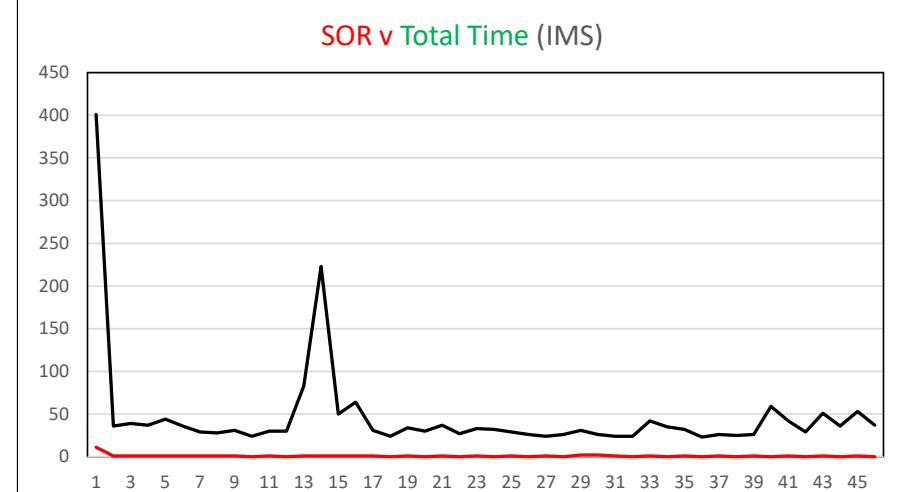
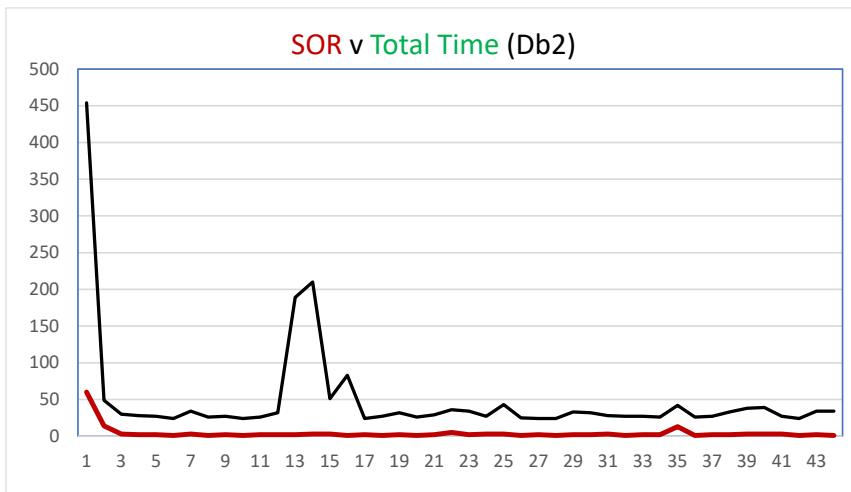
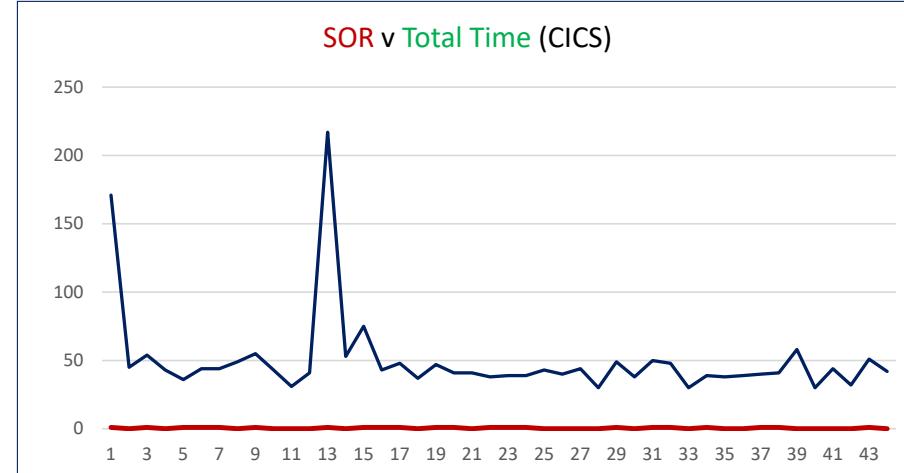
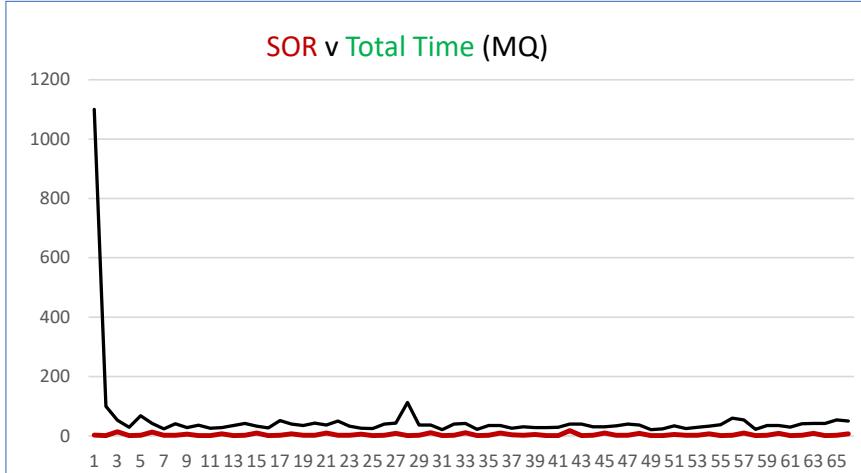
z/OS Connect SMF 123 subtype 1 version 2 (OpenAPI 2) *



Some fields have been hidden

* Generated by using a modified version of the BAQSMFX sample program.

z/OS Connect SMF 123 subtype 1 version 2 graph examples (OpenAPI 2)



z/OS Connect SMF 123 subtype 2 version 2 (OpenAPI 2) *



smfout.csv

2021/08/23 18:16:02.725340 UTC

SMF123_RSMF123_S SMF123_SUBTYPE_VERSION																												
27	123	2	2																									
30	SID	SSI	TRIPLET_C	TRIPLET_C	HTTP_REQ_STAT	REQ_RET	REQ_PAYL	RESP_PA1	USER_NA	USER_NA	ENDPOINT_I	ENDPOINT	TIME_ST	TIME_TIME_I	TII	TIME_ENPOI	StubTime	ZCInboun	TokenTim	EndPointTime	ZCOutbou	TotalTime(us)	TotalTime(s)	MVS_JOB	MVS_JOB	MVS_JOB		
31	MPZ3	ZCON	2	40	200	200	NO	0	272	USER1	GET	2021/08/2021/0220202:2021/08/2318:	95384	108577	6734453	131423	25653	7103301	7,103301	7,103301	7,103301	7,103301	7,103301	7,103301	7,103301	7,103301		
32	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	114313	7767	318	40583	2105	166276	166276	166276	166276	166276	166276	166276	166276	166276		
33	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	112903	7193	130	51158	1905	175644	175644	175644	175644	175644	175644	175644	175644	175644		
34	MPZ3	ZCON	2	40	200	200	NO	0	271	USER1	GET	2021/08/2021/0220202:2021/08/2318:	103999	102634	8843582	110850	3497	9166156	9,166156	9,166156	9,166156	9,166156	9,166156	9,166156	9,166156	9,166156		
35	MPZ3	ZCON	2	40	200	200	NO	0	271	USER1	GET	2021/08/2021/0220202:2021/08/2318:	82840	4956	128	65685	1900	156097	156097	156097	156097	156097	156097	156097	156097	156097		
36	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	116458	10778	288	58698	1778	189030	189030	189030	189030	189030	189030	189030	189030	189030		
37	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	149159	20483	614	102698	1760	277114	277114	277114	277114	277114	277114	277114	277114	277114		
38	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	153803	23181	285	101022	1775	281176	281176	281176	281176	281176	281176	281176	281176	281176		
39	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	140685	70595	11275606	113382	1920	11603168	11,603168	11,603168	11,603168	11,603168	11,603168	11,603168	11,603168	11,603168	11,603168	
40	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	108088	7624	222	65726	1746	184303	184303	184303	184303	184303	184303	184303	184303	184303		
41	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	119784	9945	282	76225	1773	209052	209052	209052	209052	209052	209052	209052	209052	209052		
42	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	94511	5061	132	44576	2427	147407	147407	147407	147407	147407	147407	147407	147407	147407		
43	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	56951	10497	126	118293	1703	189186	189186	189186	189186	189186	189186	189186	189186	189186		
44	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	55110	7646	210	122479	1616	187974	187974	187974	187974	187974	187974	187974	187974	187974		
45	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	119104	10588	354	109467	1604	242675	242675	242675	242675	242675	242675	242675	242675	242675		
46	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	3051028	17103	9999318	222997	1770	13292831	13,292831	13,292831	13,292831	13,292831	13,292831	13,292831	13,292831	13,292831	13,292831	
47	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	129965	20381	121	212563	1870	366316	366316	366316	366316	366316	366316	366316	366316	366316		
48	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	117036	17792	768	221666	1796	360790	360790	360790	360790	360790	360790	360790	360790	360790		
49	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	121667	23095	468	217285	1673	366393	366393	366393	366393	366393	366393	366393	366393	366393		
50	MPZ3	ZCON	2	40	200	200	NO	0	269	USER1	GET	2021/08/2021/0220202:2021/08/2318:	115629	13252	685	146376	1659	279825	279825	279825	279825	279825	279825	279825	279825	279825		
51																												
52	REC_TYPE	SUBTYPE	SUBTYPE	SUBTYPE	VERSION																							

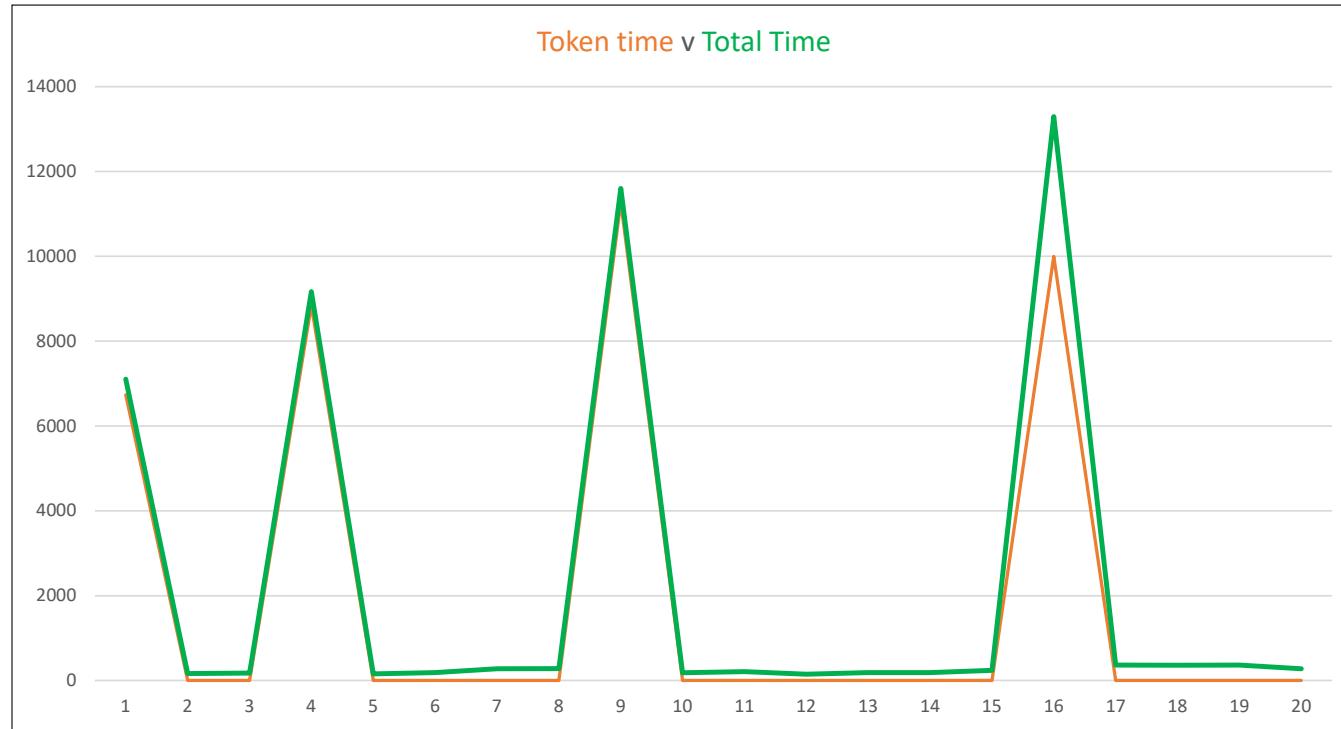
Some fields have been hidden

* Generated by using a modified version of the BAQSMFX sample program.

mitchj@us.ibm.com

© 2017, 2023 IBM Corporation
Slide 249

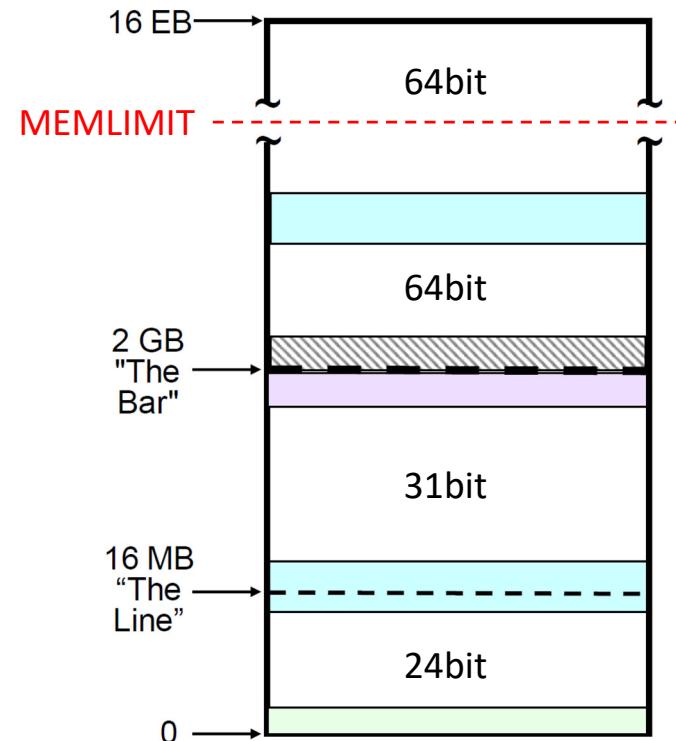
z/OS Connect SMF 123 subtype 2 version 2 graph example (OpenAPI 2)



Memory - MEMLIMIT

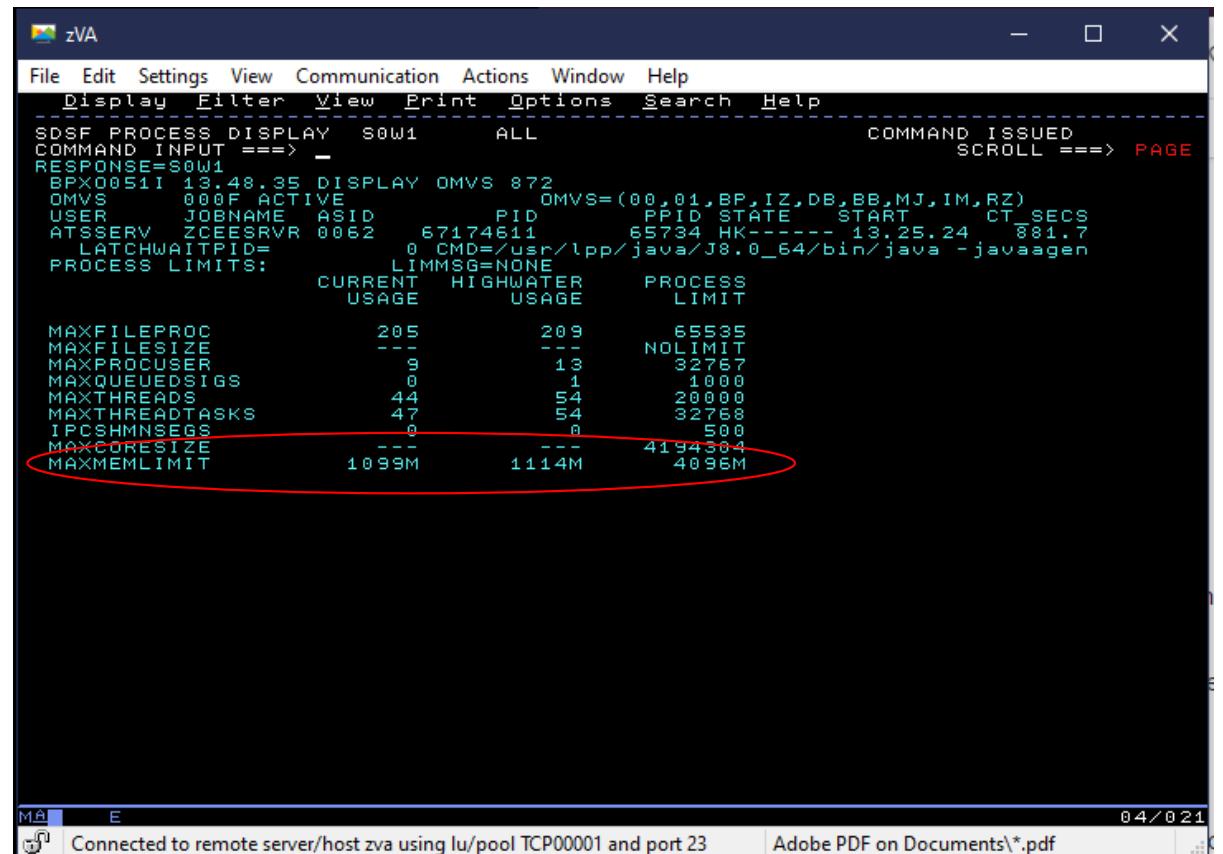
```
//ZCON EXEC PGM=BPXBATSL,REGION=0M,MEMLIMIT=8G,  
//      PARM='PGM &ZCONHOME./bin/zosconnect run &PARMS.'
```

- Limits the amount of 64-bit storage
 - Only a limit, not pre-allocated
- Java
 - Heap
 - Caches
- z/OS
 - Native thread stack storage
 - 3MB for each thread



MEMLIMIT

- OMVS display
 - Monitor periodically
 - Track high water mark
 - `/D OMVS,LIMITS,PID=<server pid>`



```

zVA
File Edit Settings View Communication Actions Window Help
Display Filter View Print Options Search Help
SDSF PROCESS DISPLAY S0W1 ALL COMMAND ISSUED
COMMAND INPUT ===> - SCROLL ===> PAGE
RESPONSE=S0W1
BPX051I 13.48.35 DISPLAY OMVS 872
OMVS 000F ACTIVE OMVS=(00,01,BP,IZ,DB,BB,MJ,IM,RZ)
USER JOBNAME ASID PID PPID STATE START CT SECS
ATSSERV ZCEESRVR 0062 67174611 65734 HK---- 13.25.24 881.7
LATCHWAITPID= 0 CMD=/usr/lpp/java/J8.0_64/bin/java -javaagen
PROCESS LIMITS: LIMMSG=NONE
               CURRENT HIGHWATER PROCESS
               USAGE   USAGE   LIMIT
MAXFILEPROC    205     209     65535
MAXFILESIZE   ---     ---     NOLIMIT
MAXPROCUSER    9       13      32767
MAXQUEUEDSIGS 0       1       1000
MAXTHREADS    44      54      20000
MAXTHREADDTASKS 47      54      32768
IPCSHMNSEGS   0       0       500
MAXCORESIZE   ---     ---     4194304
MAXMEMLIMIT    1098M   1114M   4096M

```

The screenshot shows the z/OS zVA window with the SDSF PROCESS DISPLAY command. The output lists various system parameters and their current usage versus limits. A red oval highlights the last row, which shows the maximum memory limit (MAXMEMLIMIT) set at 4096M.



MEMLIMIT Recommendations

- Don't reach the maximum!
 - Results in Java Out Of Memory errors and system abends
 - z/OS Connect EE will stop processing API requests
- Ensure this doesn't happen
 - Limit the Liberty Default Executor thread pool
 - maxThreads default value is **-1** No Limit!
 - **MEMLIMIT** =
 - Maximum JVM Heap Size (-Xmx)
 - + 20% of the Maximum Heap Size (for JIT caches and other JVM requirements)
 - + Default Executor pool maxThreads * 3MB

```
<executor maxThreads="300" />
```

Maximum JVM Heap Size – half the available memory with a minimum of 16 MB and a maximum of 512 MB



MEMLIMIT Recommendations

- Monitor thread usage for the address space
 - `/D OMVS,LIMITS,PID=<server pid>`

```
WG31 - 3270
File Edit Settings View Communication Actions Window Help
Display Filter View Print Options Search Help
SDSF PROCESS DISPLAY WG31 ALL COMMAND ISSUED
COMMAND INPUT ===> _ SCROLL ===> PAGE
RESPONSE=WG31
BPX0051I 11.14.07 DISPLAY OMVS 705
OMVS 000F ACTIVE OMVS=(Z3,MJ)
USER JOBNAM ASID PID PPID STATE START CT_SECS
LIBSERV BAQSTRT 0071 33554704 16777415 HK----- 20.16.20 96.3
LATCHWAITPID= 0 CHD=/usr/lpp/java/J8.0_64/bin/java -javaagen
PROCESS LIMITS: LIMMSG=NONE
CURRENT HIGHWATER PROCESS
USAGE USAGE LIMIT
MAXFILEPROC 203 206 10000
MAXFILESIZE -- -- NOLIMIT
MAXPROCUSER 0 7 200
MAXQUEUEDSIGS 0 1 1000
MAXTHREADS 34 40 10000
MAXTHREADTASKS 34 40 5000
LPCSHNSEGS 0 0 500
MAXCORESIZE -- -- 4194304
MAXMEMLIMIT 1026M 1061M 4096M
```

MA A 04 / 021
Connected to remote server/host wg31a using lu/pool TCP00109 and port 23 Adobe PDF on Documents*.pdf

- Ensure SOR connections are configured appropriately
 - IPIC Send Sessions, IMS Connection Pool, Db2 http max connections
- Take action when USAGE comes within 80-90% of **maxThreads**

Today we covered

- **A Review OMVS, Liberty and RACF security/configuration**
- **Connecting z/OS Connect servers to other z/OS subsystems**
- **Useful Liberty features and MVS commands**
- **Where do look when things go wrong**
- **Managing and Monitoring Liberty and z/OS Connect**
- **Additional Material - sample administrative JCL**



z/OS Connect Wildfire Github Site <https://ibm.biz/BdPRGD>

The screenshot displays two GitHub repository pages side-by-side. The left repository, titled 'ibm-wsc / zCONNEE-Wildfire-Workshop', shows a file structure with several folders and files. Two specific folders, 'AdminSecurity' and 'xml', are circled in red. The right repository, also titled 'ibm-wsc / zCONNEE-Wildfire-Workshop', shows a list of files added via upload, including various customization PDFs and an 'admin' file.

Left Repository (ibm-wsc / zCONNEE-Wildfire-Workshop):

- Code (master branch, 1 branch, 0 tags)
- Issues
- Pull requests
- Actions
- Projects
- Wiki
- Security
- Insights
- Settings

Right Repository (ibm-wsc / zCONNEE-Wildfire-Workshop):

- Code (master branch, 1 branch, 0 tags)
- Issues
- Pull requests
- Actions
- Projects
- Wiki
- Security
- Insights
- Settings

File List:

- emitchj Delete ZCONNEE - Introduction
- AdminSecurity (circled in red)
- OpenAPI2
- rcobal (circled in red)
- xml (circled in red)
- README.md
- ZADMIN - zOS Connect Administrat...
- ZEESEC - zOS Connect Security.pdf
- ZCINTRO - Introduction to zOS Conn...
- zOS Connect EE V3 Advanced Topics ...
- zOS Connect EE V3 Getting Started.pdf
- README.md

This repository contains material from

File History (Right Repository):

File	Action	Date
Customization Basic Configuration(1of2) (1).pdf	Add files via upload	last month
Customization Basic Configuration(1of2) (2).pdf	Add files via upload	last month
Customization Security and CICS.pdf	Add files via upload	last month
Customization Security and DB2.pdf	Add files via upload	last month
Customization Security and JWT Tokens.pdf	Add files via upload	last month
Customization Security and MQ.pdf	Add files via upload	last month
Customization Security when accessing an IMS Database....	Add files via upload	last month
Customization Security when accessing an IMS Transactio...	Add files via upload	last month
Customization Security with MVS Batch.pdf	Add files via upload	last month
admin	Create admin	last month

mitchj@us.ibm.com

- Contact your IBM representative to schedule access to these exercises

© 2018, 2022 IBM Corporation

Page 256

WSC wants your
feedback!

What you will see:

From: IBM Client Feedback <ibm@feedback.ibm.com> **Subject:** Got a minute? Two questions on your IBM Z Washington Systems Center experience



Dear

Thank you for engaging with our team. At IBM Z Washington Systems Center, we make it a priority to listen to our clients and want to continuously improve your experience. So, we would love your candid feedback on how we are doing. Please take a moment to answer two short questions about your experience.

You can begin the survey by answering this question.

How likely are you to recommend IBM Z Washington Systems Center to others?



Sincerely,

IBM Advocacy Team

****you will NOT receive a new survey if you already responded to an IBM Survey from Medallia in the last **60 days** OR if you haven't responded within the last **30 days******



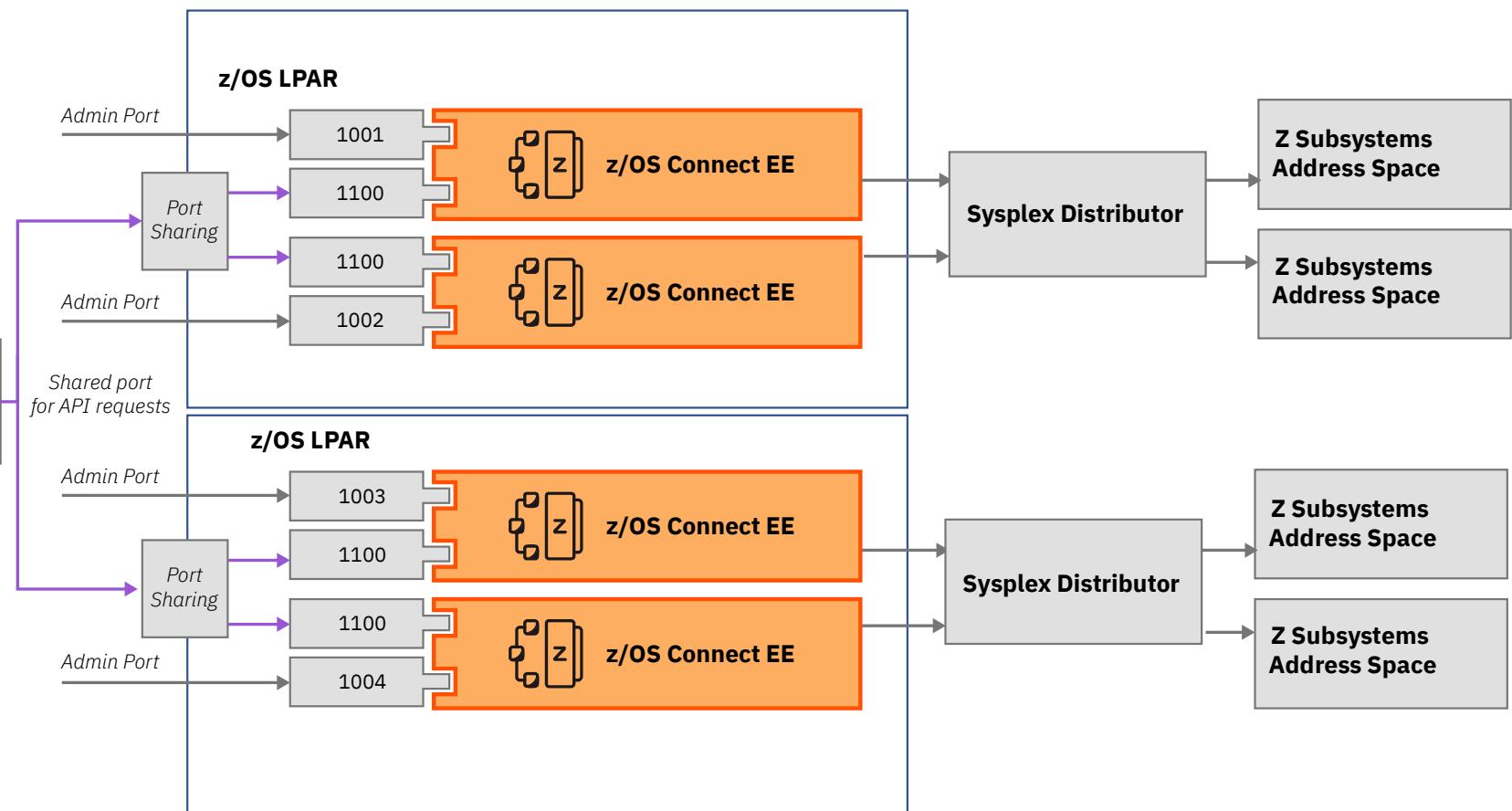
Thank you for listening and your questions.

Miscellaneous Odds and Ends



High Availability

- Topology



i ibm.biz/zosconnect-ha-concepts

i ibm.biz/zosconnect-scenarios

Sysplex DVIPAs



SYS1.TCPIP.TCPPARMS (IPNODES)

```
192.168.17.241 MPZ1.DMZ MPZ1 mpz1.washington.ibm.com  
192.168.17.242 MPZ2.DMZ MPZ2 mpz2.washington.ibm.com  
192.168.17.243 MPZ3.DMZ MPZ3 mpz3.washington.ibm.com  
192.168.17.240 dvipa dvipa.washington.ibm.com
```

SYS1.TCPIP.TCPPARMS (PROFMPZ3)

```
IPCONFIG SYSPLEXROUTING  
DYNAMICXCF 172.1.1.243 255.255.255.0 3  
VIPADYNAMIC  
VIPADEFINE 255.255.255.0 192.168.17.240  
VIPADISTRIBUTE DEFINE DISTM ROUNDROBIN|BASEWLM 192.168.17.240  
PORT 23 1416 1491 2446 9443 9453 9463  
DESTIP  
172.1.1.241  
172.1.1.242  
172.1.1.243  
ENDVIPADYNAMIC
```

SERVERWLM is not an option

HOMETEST

```
EZA0619I Running IBM MVS TCP/IP CS V2R4 TCP/IP Configuration Tester  
EZA0602I TCP Host Name is: MPZ3  
  
EZA0605I Using Name Server to Resolve MPZ3  
EZA0611I The following IP addresses correspond to TCP Host Name: MPZ3  
EZA0612I 192.168.17.243  
EZA0614I The following IP addresses are the HOME IP addresses defined in PROFILE.TCPIP:  
EZA0615I 192.168.17.243  
EZA0615I 172.1.1.243  
EZA0615I 192.168.17.240  
EZA0615I 127.0.0.1  
  
EZA0618I All IP addresses for MPZ3 are in the HOME list!  
EZA0622I Hometest was successful - all Tests Passed!
```

```
<zosconnect_cicsIpicConnection id="cscvinc"  
host="dvipa.washington.ibm.com"  
port="1491"/>  
<zosconnect_endpointConnection id="mqapi"  
host="http://dvipa.washington.ibm.com"  
port="9453"  
basicAuthRef="myBasicAuth"  
connectionTimeout="10s"  
receiveTimeout="20s" />
```

The screenshot shows a browser window displaying the Liberty REST API Documentation. The URL <https://dvipa.washington.ibm.com:9443/api/explorer/> is visible in the address bar, with a red circle highlighting it. The page itself is titled "Liberty REST APIs" and shows a list of available endpoints categorized by service name. Each service entry includes methods (POST, DELETE, GET, PUT) and their corresponding URLs.

Service	Method	URL	Operations
cscvinc	POST	/cscvinc/employee	Show/Hide List Operations Expand Operations
cscvinc	DELETE	/cscvinc/employee/{employee}	
cscvinc	GET	/cscvinc/employee/{employee}	
cscvinc	PUT	/cscvinc/employee/{employee}	
db2employee			Show/Hide List Operations Expand Operations
filemgr			Show/Hide List Operations Expand Operations
imsPhoneBook			Show/Hide List Operations Expand Operations
jwtvpDemoApi			Show/Hide List Operations Expand Operations
miniloancics			Show/Hide List Operations Expand Operations
mqapi			Show/Hide List Operations Expand Operations
phonebook			Show/Hide List Operations Expand Operations



Use z/OS Connect API Policies to change runtime behavior (OpenAPI 2)

- HTTP header properties can be used to select alternative for IMS (V3.0.4) , CICS (V3.0.10), Db2 (V3.0.36) or MQ (V3.0.39)
- Policies can be configured globally for every API in the server or for individual APIs (V3.0.11)

CICS attributes

- cicsCcsid
- cicsConnectionRef
- cicsTransId

IMS attributes

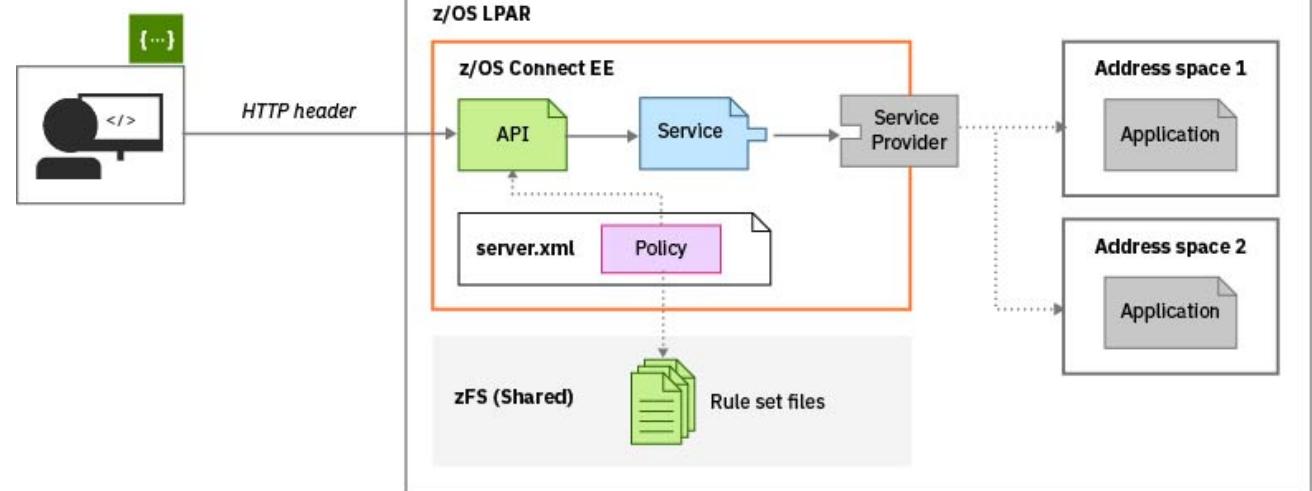
- imsConnectionRef
- imsInteractionRef
- imsInteractionTimeout
- imsLtermOverrideName
- imsTranCode
- imsTranExpiration

Db2 attributes

- db2ConnectionRef
- db2CollectionID

MQ attributes

- mqConnectionFactory
- mqDestination
- mqReplyDestination





A sample API Policies for CICS (OpenAPI 2)

```
<ruleset name="CICS rules">
  <rule name="csmi-rule">
    <conditions>
      <header name="cicsMirror" value="CSMI,MIJO"/> *
    </conditions>
    <actions>
      <set property="cicsTransId" value="${cicsMirror}"/>
    </actions>
  </rule>
  <rule name="connection-rule">
    <conditions>
      <header name="cicsConnection"
             value="cscvinc,cics92,cics93"/>
    </conditions>
    <actions>
      <set property="cicsConnectionRef" value="${cicsConnection}">
    </actions>
  </rule>
</ruleset>
```

The screenshot shows the API Designer interface for a policy named 'GET.employee.{numb}'. The policy structure is as follows:

- Body - cscvincServiceOperation**: Contains a single parameter **cscvincServiceOperation**.
- HTTP Request**: Contains:
 - HTTP Headers**: Contains two parameters:
 - cicsMirror**: optional string
 - cicsConnection**: optional string
 - Path Parameters**: Contains one parameter:
 - numb**: Required string
 - Query Parameters**: Contains a single parameter:
 - Body - cscvincServiceOperation**

Curl

```
curl -X GET --header 'Accept: application/json' --header 'cicsMirror: MIJO' --header 'cicsConnection: cscvinc' 'https://m...
```

*Transaction MIJO needs to be a clone of CSMI (e.g., invoke program DFHMIRS)



Displaying zCEE messages on the console and/or STDERR spool

server.xml

```
<zosLogging wtoMessage=
  "BAQR0657E,BAQR0658E,BAQR0660E,BAQR0686E,BAQR0687E"
  hardCopyMessage=
  "BAQR0657E,BAQR0658E,BAQR0660E,BAQR0686E,BAQR0687E"/>
```

MVS Console

```
18.12.02 STC00137 +BAQR0686E: Program CSCVINC is not available in the CICS region with
  811           connection ID cscvinc; service cscvincService failed.
18.12.02 STC00137 +BAQR0686E: Program CSCVINC is not available in the CICS region with
  812           connection ID cscvinc; service cscvincService failed.
19.07.12 STC00137 +BAQR0657E: Transaction abend MIJO occurred in CICS while using
  745           connection cscvinc and service cscvincService.
```

STDERR

```
ÝERROR  `` BAQR0686E: Program CSCVINC is not available in the CICS region with connection cscvinc and service cscvincService.
ÝERROR  `` BAQR0686E: Program CSCVINC is not available in the CICS region with connection cscvinc and service cscvincService.
ÝERROR  `` BAQR0657E: Transaction abend MIJO occurred in CICS while using CICS connection cscvinc and service cscvincService.
```

Additional Material

Sample Administrative JCL and other topics

Sample JCL - Check Java installation by displaying Java version information

```
//JOHNSONS JOB (ACCOUNT),JOHNSON,NOTIFY=JOHNSON,REGION=0M,  
// CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),USER=LIBSERV  
//*****  
//* SET SYMBOLS  
//*****  
//EXPORT EXPORT SYMLIST=(*)  
// SET JAVAHOME='/usr/lpp/java/J8.0_64'  
//*****  
//* STEP JAVA - INVOKE THE java -version COMMAND  
//*****  
//JAVA EXEC PGM=IKJEFT01,REGION=0M  
//SYSTSPRT DD SYSOUT=*  
//SYSERR DD SYSOUT=*  
//STDOUT DD SYSOUT=*  
//STDENV DD DUMMY  
//SYSTSIN DD *,SYMBOLS=EXECSYS  
BPXBATCH SH +  
export JAVA_HOME=&JAVAHOME; +  
$JAVA_HOME/bin/java -version
```

Requires RACF SUROGAT access

Sample JCL - Executing the z/OS Connect zconsetup script

```
//JOHNSONS JOB (ACCOUNT),JOHNSON,NOTIFY=JOHNSON,REGION=0M,  
// CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1)  
//*****  
//* Set symbols  
//*****  
//EXPORT EXPORT SYMLIST=(*)  
// SET JAVAHOME='/usr/lpp/java/J8.0_64'  
// SET ZCEEPATH='/usr/lpp/IBM/zosconnect/v3r0'  
//*****  
//* Step ZCSETUP - Invoke the zconsetup script  
//*****  
//ZCSETUP EXEC PGM=IKJEFT01,REGION=0M  
//SYSTSPRT DD SYSOUT=*  
//SYSERR DD SYSOUT=*  
//STDOUT DD SYSOUT=*  
//SYSTSIN DD *,SYMBOLS=EXECSYS  
BPXBATCH SH +  
  export JAVA_HOME=&JAVAHOME; +  
  &ZCEEPATH/bin/zconsetup install
```

Sample JCL - Executing the z/OS Connect Build Toolkit on z/OS



```
//JOHNSONS JOB (ACCOUNT),JOHNSON,NOTIFY=&SYSUID,REGION=0M,  
// CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1)  
//*****  
///* SET SYMBOLS  
//*****  
//EXPORT EXPORT SYMLIST=(*)  
// SET WORKDIR='u/johnson/zconbt'  
// SET ZCONDIR='/usr/lpp/IBM/zosconnect/v3r0/zconbt/bin'  
//ZCONBT EXEC PGM=IKJEFT01,REGION=0M,MEMLIMIT=4G  
//SYSTSPPRT DD SYSOUT=*  
//SYSERR DD SYSOUT=*  
//STDOUT DD SYSOUT=*  
//SYSTSIN DD *,SYMBOLS=EXECSYS  
BPXBATCH SH +  
  export WORKDIR=&WORKDIR; +  
  export ZCONDIR=&ZCONDIR; +  
  cd $WORKDIR; +  
  $ZCONDIR/zconbt.zos -p cscvinc.properties -f=cscvinc.ara; +  
  cp -v $WORKDIR/syslib/* //'JOHNSON.ZCONBT.COPYLIB'"
```

cscvinc.properties

```
apiDescriptionFile=./cscvinc.json  
dataStructuresLocation=./syslib  
apiInfoFileLocation=./syslib  
logFileDirectory=./logs  
language=COBOL  
connectionRef=cscvincAPI  
requesterPrefix=csc
```

This assumes the zconbt.zip files was expanded into directory /usr/lpp/IBM/zosconnect/v3r0/zconbt using command *jar -tf zconbt.zip* and that the property file and Swagger JSON document are encoded in ASCII in directory /u/johnson/zconbt.

Server XML – Accessing a HATS REST service (OpenAPI 2)



```
getCompany.properties - Notepad
File Edit Format View Help
provider=rest
name=getCompany
version=1.0
description=Obtain a list of companies
requestSchemaFile=getCompanyRequest.json
responseSchemaFile=getCompanyResponse.json
verb=POST
uri=/Trader/rest/GetCompany
connectionRef=HatsConn
```

Server Config

hats.xml

Read only Close

Design Source

```
<server description="HATS">
  <zosconnect_zosConnectServiceRestClientConnection id="HatsConn"
    host="wg31.washington.ibm.com"
    port="29080" />
</server>
```

HATS Liberty server.xml

```
<!-- To access this server from a remote client, add a host attribute to the following element, e.g. host="*" -->
<httpEndpoint id="defaultHttpEndpoint"
  httpPort="29080"
  host="*"
  httpsPort="29443" />
```

Server XML- Accessing an MVS application using WOLA (OpenAPI 2)

```
filea.properties - Notepad
File Edit Format View Help
name=Filea
version=1.0
provider=wola
description=Test COBOL batch program
language=COBOL
program=ATSFIL
register=FILEAZCON
connectionRef=wolaCF
requestStructure=./fileareq.cpy
responseStructure=./filearsp.cpy
```

```
Server Config
wola.xml
Read only Close
Design Source
<server description="WOLA">
  <featureManager>
    <feature>zosLocalAdapters-1.0</feature>
  </featureManager>
  <zosLocalAdapters wolaGroup="ZCEESRVR">
    <wolaName2="ZCEESRVR"/>
    <wolaName3="ZCEESRVR"/>
  </zosLocalAdapters>
  <connectionFactory id="wolaCF">
    <jndiName="eis/ola">
      <properties.ola/>
    </connectionFactory>
  </connectionFactory>
</server>
```

```
* SET THE VALUES FOR USE WITH WOLA REGISTRATION
MOVE 'FILEAZCON'          TO REG-REGNAME.
MOVE 'ZCEESRVR'            TO REG-DAEMONGRP.
MOVE 'ZCEESRVR'            TO REG-NODE.
MOVE 'ZCEESRVR'            TO REG-SVRNAME.
MOVE 'ATSFIL'              TO SVC-SERVICE-NAME.
INSPECT REG-DAEMONGRP CONVERTING ' ' to LOW-VALUES.
* Register to a Local Liberty server
CALL 'BBOA1REG' USING
  REG-DAEMONGRP,REG-NODE,REG-SVRNAME,REG-REGNAME,REG-MINCONN,REG-MAXCONN,REG-FLAGS,RSP-RC,RSP-RSN.
```



Server XML – Accessing a DVM server using WOLA (OpenAPI 2)

Server Config

dvs.xml

Read only Close

Design Source

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="new server">
  <!-- Enable features -->
  <featureManager>
    <feature>usr:dvsProvider</feature>
    <feature>zosLocalAdapters-1.0</feature>
  </featureManager>
  <!-- Adapter Details with WOLA Group Name (ZCEEDVM) -->
  <zosLocalAdapters wolaName3="NAME3"
    wolaName2="NAME2"
    wolaGroup="ZCEEDVM"/>
  <!-- DVS Service Details with Register Name (ZCEEDVM) -->
  <zosconnect_zosConnectService invokeURI="/dvs">
    serviceDescription=""
    serviceRef="dvsService"
    serviceName="dvsService"
    id="zosConnectDvsService"/>
  <usr_dvsService invokeURI="/dvs">
    serviceName="DVSS1"
    registerName="ZCEEDVM"
    connectionFactoryRef="wolaCF"
    id="dvsService"/>
  <connectionFactory jndiName="eis/ola" id="wolaCF">
    <properties.ola/>
  </connectionFactory>
  <zosconnect_zosConnectService serviceRef="svc1">
    serviceAsyncRequestTimeout="600s"
    serviceName="dvs1" id="sdef1"/>
  <zosconnect_localAdaptersConnectService
    connectionWaitTimeout="7200"
    connectionFactoryRef="wolaCF"
    serviceName="DVSS1"
    registerName="ZCEEDVM"
    id="svc1"/>
</server>
```

DVS.AVZS.SAVZEXEC (AVZSIN00)

```
/*
 * Enable z/OS Connect interface facility
 */
if DoThis then
  do
    /*
     * The following parameter enables the z/OS Connect interface
     * facility.
    */
    "MODIFY PARM NAME(ZCONNECT)           VALUE(YES)"
    "MODIFY PARM NAME(NETWORKBUFFERSIZE)   VALUE(96K)"
  /*
   * The "DEFINE ZCPATH" command(s) can be used to define
   * paths to z/OS Connect regions to handle requests.
   * Use a separate "DEFINE ZCPATH" command to define each
   * path required (Note that a single path can handle
   * several different requests)
   * refer to the documentation for details about the parameters,
   * and information about optional parameters.
  */
  "DEFINE ZCPATH",
    "  NAME(ZCEE)                      ",
    "  RNAME(ZCEEDVM)                  ",
    "  WNAME(ZCEEDVM)                  ",
    ""
end
```

Server XML – Accessing a File Manager server (OpenAPI 2)

```
filea.properties - Notepad
File Edit Format View Help
name=filea
provider=filemanager
host=wg31.washington.ibm.com
version=1.0
port=2800
file=USER1.ZCEE.FILEA
template=USER1.ZCEE.TEMPLATE(FILEA)
connid=default
userid=USER1
passwd=USER1

Ln 1, Col 1 100% Windows (CRLF) UTF-8
```

Server Config

filemgr.xml

Design Source

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="new server">
  <!-- Enable features -->
  <featureManager>
    <feature>filemanager:fmProvider-2.0</feature>
  </featureManager>
  <FileManager_Connection id="default">
    <runport>2800</runport>
    <max_timeout>1800</max_timeout>
  </FileManager_Connection>
</server>
```

Read only Close

SYS1.PROCLIB(IPVSRV1)

```
//IPVSRV1 PROC PORT=2800,FAMILY='AF_INET',TRACE=N
//      SET ENV=''
//RUN      EXEC PGM=IPVSRV,REGION=40M,
//      PARM='(&ENV/&PORT &FAMILY &TRACE')
// SET IPV=SYSP.ADFZ.JCL          <== Update HLQ
//STEPLIB  DD DISP=SHR,DSN=ADFZ.SIPVMODA      <== ADFzCC APF LIBRARY
//SYSPRINT DD SYSOUT=*
//IPVTRACE DD SYSOUT=*
//STDOUT   DD SYSOUT=*
///* Server wide, then participating product configurations
//CONFIG   DD DISP=SHR,DSN=&IPV.(IPVCFG)
```



Tech-Tip: Liberty's “adminCenter” Feature

- The Web browser interface feature “adminCenter” was used to display the server’s configuration files

The screenshot shows two side-by-side views of the IBM Liberty adminCenter interface. Both views are for the 'server.xml' configuration file.

The left view is the 'Design' tab, which displays a tree-based configuration editor. The tree shows several 'Include' statements pointing to various XML files under '/var/zosconnect/servers/myServer/resources'. A 'Server' node is selected, and its configuration details are shown below it, including an 'Add child' button and a 'Remove' button. The 'Description' field contains the value 'new server'.

The right view is the 'Source' tab, which displays the raw XML code for the 'server.xml' file. The code includes declarations for 'imsmobile-config/services/ims-services.xml', 'imsmobile-config/interactions/ims-interactions.xml', 'imsmobile-config/connections/ims-connections.xml', and 'imsmobile:ims-admin-services.xml'. It also includes sections for 'saFSecurity.xml', 'saFTrace.xml', 'ipic.xml', 'keyring.xml', 'apiRequesterHTTPS.xml', 'shared.xml', 'oauth.xml', 'audit.xml', 'mq.xml', 'db2.xml', 'wlm.xml', 'restConnector.xml', 'smf.xml', and 'adminCenter.xml'. A note at the bottom indicates that a 'host' attribute can be added to the 'httpEndpoint' element to access the server from a remote client.

Tech/Tip: Use the TCPIP resolver trace to display name resolution information

```
ALLOC FILE(SYSTCPT) DA(*)  
ping wg31.washington.ibm.com  
Resolver Trace Initialization Complete -> 2021/09/12 12:54:37.36  
  
res_init Resolver values:  
Setup file warning messages = No  
CTRACE TRACERES option = No  
Global Tcp/Ip Dataset = SYS1.TCPIP.TCPPARMS(TCPDAT3)  
Default Tcp/Ip Dataset = SYS1.TCPIP.TCPPARMS(TCPDAT3)  
Local Tcp/Ip Dataset = //DD:SYSTCPD  
                      ==> SYS1.TCPIP.TCPPARMS(TCPDAT3)  
Translation Table = SYS1.TCPIP.STANDARD.TCPXLBIN  
UserId/JobName = JOHNSON  
Caller API = TCP/IP Sockets Extended  
Caller Mode = EBCDIC  
System Name = WSC13 (from VMCF)  
UnresponsiveThreshold = 25  
(G) DataSetPrefix = SYS1.TCPIP  
(G) HostName = MPZ3  
.  
.  
.  
res_query Failed: RetVal = -1, RC = 1, Reason = 0x78981005  
res_querydomain Failed: RetVal = -1, RC = 1, Reason = 0x78981005  
res_search Failed: RetVal = -1, RC = 1, Reason = 0x78981005  
GetAddrInfo Closing IOCTL Socket 0x00000000  
BPX1CLO: RetVal = 0, RC = 0, Reason = 0x00000000  
GetAddrInfo Failed: RetVal = -1, RC = 1, Reason = 0x78AE1004  
GetAddrInfo Ended: 2021/09/12 12:55:32.364732  
*****  
EZZ3111I Unknown host 'WG31.WASHINGTON.IBM.COM'
```

Root cause – Host wg31.washington.ibm.com was missing from SYS1.TCPIP.TCPPARMS(IPNODES)



Sample JCL - Executing the Liberty *securityUtility* command

```
*****  
/* Use securityUtility to encrypt a password using an  
/* encryption key of a certificate  
*****  
//IKJEFT01 EXEC PGM=IKJEFT01,REGION=0M  
//SYSTSPRT DD SYSOUT=*  
//SYSERR DD SYSOUT=*  
//STDOUT DD SYSOUT=*  
//SYSTSIN DD *  
BPXBATCH SH +  
/usr/lpp/IBM/zosconnect/v3r0/wlp/bin/securityUtility encode +  
--encoding=aes +  
--keyring=safkeyring://JOHNSON/Liberty.KeyRing +  
--keyringType=JCERACFKS --keyLabel="Johnson Client Cert" +  
passwordToEncrypt
```

```
<featureManager>  
  <feature>zosPasswordEncryptionKey-1.0</feature>  
</featureManager>  
  
<zosPasswordEncryptionKey  
keyring="safkeyring://JOHNSON/Liberty.KeyRing"  
label="Johnson Client Cert" type="JCERACFKS"/>
```

```
*****  
/* Use securityUtility to encrypt a password using an  
/* encryption key string  
*****  
//IKJEFT01 EXEC PGM=IKJEFT01,REGION=0M  
//SYSTSPRT DD SYSOUT=*  
//SYSERR DD SYSOUT=*  
//STDOUT DD SYSOUT=*  
//SYSTSIN DD *  
BPXBATCH SH +  
/usr/lpp/IBM/zosconnect/v3r0/wlp/bin/securityUtility encode +  
--encoding=aes -key myEncryptionKey +  
passwordToEncrypt
```

```
wlp.password.encryption.key=myEncryptionKey
```

Sample JCL - Executing multiple OMVS commands in one step

```

//*****
//* SET SYMBOLS
//*****
//EXPORT EXPORT SYMLIST=(*)
// SET CURL= '/usr/lpp/rocket/curl'
//*****
//* CURL Procedure
//*****
//CURL PROC
//CURL EXEC PGM=IKJEFT01,REGION=0M
//SYSTSPRT DD SYSOUT=*
//SYSERR  DD SYSOUT=*
//STDOUT   DD SYSOUT=*
// PEND
//*****
//* STEP CURL - use cURL to deploy API cscvinc
//*****
//DEPLOY EXEC CURL
BPXBATCH SH export CURL=&CURL; +
$CURL/bin/curl -X PUT -s +
--cacert /u/johnson/CERTAUTH.PEM --user FRED:FRED +
https://wg31.washington.ibm.com:9445/zosConnect/apis/cscvinc?status=sto+
pped > null; +
$CURL/bin/curl -X DELETE -s +
--cacert /u/johnson/CERTAUTH.PEM --user FRED:FRED +
https://wg31.washington.ibm.com:9445/zosConnect/apis/cscvinc > null; +
$CURL/bin/curl -X POST -s +
--cacert /u/johnson/CERTAUTH.PEM --user FRED:FRED +
--data-binary @/u/johnson/cscvinc.aar +
--header "Content-Type: application/zip" +
https://wg31.washington.ibm.com:9445/zosConnect/apis
//*****
//* STEP CURL - use cURL to invoke the API cscvinc
//*****
//INVOKE EXEC CURL
//SYSTSIN DD *,SYMBOLS=EXECSYS
BPXBATCH SH export CURL=&CURL; $CURL/bin/curl -X GET -s +
--cacert /u/johnson/CERTAUTH.PEM --user FRED:FRED +
https://wg31.washington.ibm.com:9445/cscvinc/employee/000100

```

Always be aware of the beginning and trailing spaces.

[https://www.rocketsoftware.com/
platforms/ibm-z/curl-for-zos](https://www.rocketsoftware.com/platforms/ibm-z/curl-for-zos)



Sample JCL - Executing the Liberty *productInfo* command

```
/******  
/* SET SYMBOLS  
/******  
//EXPORT EXPORT SYMLIST=(*  
// SET WLPDIR='/usr/lpp/IBM/zosconnect/v3r0/wlp'  
//PRODINFO EXEC PGM=IKJEFT01,REGION=0M,MEMLIMIT=4G  
//SYSTSPRT DD SYSOUT=*  
//SYSERR DD SYSOUT=*  
//STDOUT DD SYSOUT=*  
//SYSTSIN DD *,SYMBOLS=EXECSYS  
BPXBATCH SH +  
Export WLPDIR=&WLPDIR; +  
$WLPDIR/bin/productInfo version; +  
$WLPDIR/bin/productInfo featureInfo | grep cics; +  
$WLPDIR/bin/productInfo featureInfo | grep mq; +  
$WLPDIR/bin/productInfo featureInfo | grep ims; +  
$WLPDIR/bin/productInfo validate | grep 'Product validation'
```

```
productInfo featureInfo  
productInfo version  
productInfo validate
```

```
Product name: z/OS Connect  
Product version: 03.00.48  
Product edition: z/OS Connect Enterprise Edition
```

```
cicsService-1.0 "1.0.0"  
wmqJmsClient-1.1 "1.0.0"  
wmqJmsClient-2.0 "1.0.0"  
Product Extension: mqzosconnect  
mqService-1.0 "1.0.0"  
Product Extension: imsmobile  
imsmobile-2.0 "2.0.0.202108160933"  
Product validation completed successfully.
```

Sample JCL - Copy WOLA executables from OMVS to a PDSE

```
//JOHNSONS JOB (ACCOUNT),JOHNSON,NOTIFY=JOHNSON,REGION=0M,  
// CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1)  
//*****  
/* SET SYMBOLS  
*****  
//EXPORT EXPORT SYMLIST=(*  
// SET DSNAME='USER1.WOLA2106.LOADLIB'  
// SET ZCEEPATH='/usr/lpp/IBM/zosconnect/v3r0'  
// SET JAVAHOME='/usr/lpp/java/J8.0_64'  
//*****  
/* Step ALLOC - Allocate a PDSE load library  
*****  
//ALLOC EXEC PGM=IDCAMS  
//SYSPRINT DD SYSOUT=*  
//SYSIN DD *,SYMBOLS=EXECSYS  
DELETE '&DSNAME'  
SET MAXCC=0  
ALLOC DSNAME('&DSNAME') -  
    NEW CATALOG SPACE(2,1) DSORG(PO) CYLINDERS -  
    RECFM(U) DSNTYPE(LIBRARY)  
//*****  
/* Step WOLACOPY - copy the WOLA executables to the PDSE  
*****  
//WOLACOPY EXEC PGM=IKJEFT01,REGION=0M  
//SYSTSPRT DD SYSOUT=*  
//SYSERR DD SYSOUT=*  
//STDOUT DD SYSOUT=*  
//SYSTSIN DD *,SYMBOLS=EXECSYS  
BPXBATCH SH +  
  export JAVA_HOME=&JAVAHOME; +  
  export DSNAME=&DSNAME; +  
  cp -Xv &ZCEEPATH/wlp/clients/zos/* "//$DSNAME"
```



Sample JCL - BBOSMFV (Extract Liberty SMF 120 Subtype 11 records)

```
//JOHNSONS JOB (ACCOUNT),JOHNSON,NOTIFY=JOHNSON,REGION=0M,  
// CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1)  
//EXPORT EXPORT SYMLIST=(*)  
// SET REPORT='LibertyExport'  
//JAVA EXEC PROC=JVMPRC86,  
// JAVACLS='com.ibm.ws390.sm.smfview.JclSmf'  
//STDENV DD DISP=SHR,DSN=JOHNSON.JCLLIB.CNTL(STDENV)  
//SMFDATA DD DISP=SHR,DSN=MPZ3.DUMPSMF  
//SMFENV DD *,SYMBOLS=EXECSYS  
# Specify the plugin to use  
plugin=&REPORT  
# Specify where the output goes  
output=/u/johnson/&REPORT..csv  
# Uncomment (and change the value as appropriate) to filter  
#matchServer=BAQSTRT
```

```
JOHNSON.JCLLIB.CNTL (STDENV)  
. /etc/profile  
export JAVA_HOME=/usr/lpp/java/J8.0_64  
export PATH=/bin:"${JAVA_HOME}"/bin  
  
LIBPATH=/lib:/usr/lib:"${JAVA_HOME}"/bin  
LIBPATH="$LIBPATH":${JAVA_HOME}/lib/s390x  
LIBPATH="$LIBPATH":${JAVA_HOME}/lib/s390x/j9vm  
LIBPATH="$LIBPATH":${JAVA_HOME}/bin/classic  
export LIBPATH="$LIBPATH":  
  
# Customize your CLASSPATH here  
APP_HOME=${JAVA_HOME}  
CLASSPATH=$APP_HOME:${JAVA_HOME}/lib:${JAVA_HOME}/lib/ext  
CLASSPATH=/u/johnson/lib/bbosmfv.jar:$CLASSPATH  
CLASSPATH=/u/johnson/lib/WP102312_Plugins.jar:$CLASSPATH  
  
# Add Application required jars to end of CLASSPATH  
for i in "${APP_HOME}/*.jar; do  
    CLASSPATH="$CLASSPATH":$i"  
done  
export CLASSPATH="$CLASSPATH":  
  
# Configure JVM options  
IJO="-Xms16m -Xmx128m"  
export IBM_JAVA_OPTIONS="$IJO "
```

Sample JCL – Using ADRDSSU to dump/restore MVS data sets



ZCEEDUMP

```
//EXPORT EXEC PGM=IDCAMS  
// SET ZCEELVL=349  
//DELETE EXEC PGM=IDCAMS  
//SYSPRINT DD SYSOUT=*  
//SYSIN DD *,SYMBOLS=EXECSYS  
    DELETE IBM.ZCEE30.BKUP&ZCEELVL.  
    SET MAXCC=0  
//DUMP EXEC PGM=ADRDSU,REGION=2048K  
//SYSPRINT DD SYSOUT=*  
//DUMPDD DD DSN=IBM.ZCEE30.BKUP&ZCEELVL.,  
//           DISP=(NEW,CATLG),  
//           UNIT=SYSDA,SPACE=(CYL,(3000,2000,0),RLSE)  
//SYSIN DD *,SYMBOLS=EXECSYS  
    DUMP DATASET(INCLUDE( -  
        ZCEE30.SBAQ* -  
        ZCEE30.WOLA*.* -  
        OMVS.ZCEE*.* -  
    )) OPTIMIZE(4) OUTDDNAME(DUMPDD) TOLERATE(ENQF)
```

ZCEERSTR

```
//RESTORE EXEC PGM=ADRDSU,REGION=2048K  
//SYSPRINT DD SYSOUT=*  
//DUMPDD DD DISP=SHR,DSN=JOHNSON.ZCEE30.BKUP349  
//SYSIN DD *  
    RESTORE DATASET(INCLUDE(**)) -  
    INDDNAME(DUMPDD) OUTDYNAM(WAS004) -  
    NULLSTORCLAS -  
    REPLACE CATALOG TOLERATE(ENQF)
```

Sample JCL – Define and format a ZFS data set using IOEAGFMT

```
ZFS
//DEFINE EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//AMSDUMP DD SYSOUT=*
//SYSIN DD *
      SET MAXCC=0
      DEFINE CLUSTER (NAME(OMVS.ZCEE.GROUP1.ZFS) -
                      LINEAR CYLINDERS(100 100) SHAREOPTIONS(3))
//CREATE EXEC PGM=IOEAGFMT,REGION=0M,
// PARM=(-aggregate OMVS.ZCEE.GROUP1.ZFS -compat')
//SYSPRINT DD SYSOUT=*
//STDOUT DD SYSOUT=*
//STDERR DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//CEEDUMP DD SYSOUT=*
```

Sample JCL – Generate WLM Workload Activity Reports

```
//JOHNSONS JOB (ACCOUNT),NOTIFY=&SYSUID,REGION=0M,  
// CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1)  
//DELETE EXEC PGM=IDCAMS  
//SYSPRINT DD SYSOUT=*  
//SYSIN DD *  
    DELETE JOHNSON.DUMPSSMF.SORT  
//RMFSORT EXEC PGM=SORT,REGION=0M  
//SORTIN DD DISP=SHR,DSN=MPZ3.DUMPSSMF  
//SORTOUT DD DISP=(,CATLG),DSN=JOHNSON.DUMPSSMF.SORT,  
//           SPACE=(CYL,(100,50),RLSE),UNIT=SYSDA  
//SORTWK01 DD DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(100))  
//SORTWK02 DD DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(100))  
//SORTWK03 DD DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(100))  
//SORTWK04 DD DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(100))  
//SORTWK05 DD DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(100))  
//SORTWK06 DD DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(100))  
//SORTWK07 DD DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(100))  
//SYSPRINT DD SYSOUT=(,)  
//SYSOUT DD SYSOUT=(,)  
//SYSIN DD *  
    SORT FIELDS=(11,4,CH,A,7,4,CH,A),EQUALS  
    MODS E15=(ERBPPE15,36000,,N),E35=(ERBPPE35,3000,,N)  
//RMFPP EXEC PGM=ERBRMFPP,REGION=0M  
//SYSUDUMP DD SYSOUT=*  
//STEPLIB DD DSN=SYS1.COMBINED.LINKLIB,DISP=SHR  
//MFPIINPUT DD DISP=SHR,DSN=JOHNSON.DUMPSSMF.SORT  
//MFPMMSGDS DD SYSOUT=*  
//SYSIN DD *  
    SYSOUT(O)  
    SYSRPTS(WLMGL(RCPER)) /*WORKLOAD ACTIVITY REPORT */
```

BAQSMFP output (OpenAPI 2)



```
*****
* SMF123.1 V2 Request Data Section *
*****
SMF123S1_REQ_TYPE = API (1)
SMF123S1_HTTP_RESP_CODE = 500
SMF123S1_REQ_TIMED_OUT = NO
SMF123S1_USER_NAME = FRED
SMF123S1_USER_NAME_MAPPED =
SMF123S1_CLIENT_IP_ADDR = 192.168.0.60
SMF123S1_API_NAME = db2employee
SMF123S1_API_VERSION = 1.0.0
SMF123S1_SERVICE_NAME = selectEmployee
SMF123S1_SERVICE_VERSION = 1.0.0
SMF123S1_REQ_METHOD = GET
SMF123S1_REQ_QUERY_STR =
SMF123S1_REQ_TARGET_URI = /db2/employee/000010
SMF123S1_REQ_PAYLOAD_LEN = 0
SMF123S1_RESP_PAYLOAD_LEN = 0
SMF123S1_TIME_ZC_ENTRY = 0x00DA2FB8 38ED5494 04000000 08880001
UTC_CONV_TIME_ZC_ENTRY = 2021/08/19 15:30:24.905545 UTC
SMF123S1_TIME_ZC_EXIT = 0x00DA2FB8 38F3883F A8000000 08880001
UTC_CONV_TIME_ZC_EXIT = 2021/08/19 15:30:24.930947 UTC
SMF123S1_TIME_SOR_SENT = 0x00DA2FB8 38F232A9 76000000 08A00001
UTC_CONV_TIME_SOR_SENT = 2021/08/19 15:30:24.925482 UTC
SMF123S1_TIME_SOR_RECV = 0x00DA2FB8 38F300A4 AA000000 08880001
UTC_CONV_TIME_SOR_RECV = 2021/08/19 15:30:24.928778 UTC
SMF123S1_SP_NAME = restclient-1.0
SMF123S1_SOR_REFERENCE = Db2Conn
SMF123S1_SOR_IDENTIFIER = Db2:DSN2LOC,wg31.washington.ibm.com:2446
SMF123S1_SOR_RESOURCE = services/zCEEService/selectEmployee
SMF123S1_REQ_ID = 302
SMF123S1_TRACKING_TOKEN = 0x42415131 77734859 41514159 314E6670 31395046
35304455 312B6E7A 51454241
514E6F76 75446A74 564A5145 41413D3D 40404040 40404040 40404040
SMF123S1_REQ_HDR1 =
SMF123S1_REQ_HDR2 =
SMF123S1_REQ_HDR3 =
SMF123S1_REQ_HDR4 =
SMF123S1_RESP_HDR1 =
SMF123S1_RESP_HDR2 =
SMF123S1_RESP_HDR3 =
```

```
*****
* SMF123.2 V2 Request Data Section *
*****
SMF123S2_REQ_APP_TYPE = ZOS (3)
SMF123S2_HTTP_RESP_CODE = 200
SMF123S2_REQ_STATUS_CODE = 200
SMF123S2_REQ_RETRY = NO
SMF123S2_REQ_PAYLOAD_LEN = 0
SMF123S2_RESP_PAYLOAD_LEN = 269
SMF123S2_USER_NAME = USER1
SMF123S2_USER_NAME_MAPPED =
SMF123S2_USER_NAME_ASSERTED = USER1
SMF123S2_API_REQ_NAME = cscvinc 1.0.0
SMF123S2_API_REQ_VERSION = 1.0.0
SMF123S2_ENDPOINT_REFERENCE = cscvincAPI
SMF123S2_ENDPOINT_HOST = https://mpz3.washington.ibm.com
SMF123S2_ENDPOINT_PORT = 9463
SMF123S2_ENDPOINT_FULL_PATH = /cscvinc/employee/111111
SMF123S2_ENDPOINT_METHOD = GET
SMF123S2_ENDPOINT_STUB_STR
SMF123S2_TIME_STUB_SENT = 0x00DA2FC1 7D34CE8B 4A000000 084C0001
UTC_CONV_TIME_STUB_SENT = 2021/08/19 16:11:52.420584 UTC
SMF123S2_TIME_ZC_ENTRY = 0x00DA2FC1 7D58AE00 0E000000 08A00001
UTC_CONV_TIME_ZC_ENTRY = 2021/08/19 16:11:52.567534 UTC
SMF123S2_TIME_ZC_EXIT = 0x00DA2FC1 87DCB806 E6000000 08880001
UTC_CONV_TIME_ZC_EXIT = 2021/08/19 16:12:03.594112 UTC
SMF123S2_TIME_TOKEN_GET_START = 0x00DA2FC1 7D59D3A6 E6000000 08A00001
UTC_CONV_TIME_TOKEN_GET_START = 2021/08/19 16:11:52.572218 UTC
SMF123S2_TIME_TOKEN_GET_FINISH = 0x00DA2FC1 7D59DF85 CC000000 088C0001
UTC_CONV_TIME_TOKEN_GET_FINISH = 2021/08/19 16:11:52.572408 UTC
SMF123S2_TIME_ENDPOINT_SENT = 0x00DA2FC1 7D5A0328 04000000 088C0001
UTC_CONV_TIME_ENDPOINT_SENT = 2021/08/19 16:11:52.572978 UTC
SMF123S2_TIME_ENDPOINT_RECEIVED = 0x00DA2FC1 87DCB816 58000000 08880001
UTC_CONV_TIME_ENDPOINT_RECEIVED = 2021/08/19 16:12:03.593249 UTC
SMF123S2_MVS_JOBNAME = USER1GE2
SMF123S2_MVS_JOBID = JOB09543
SMF123S2_MVS_SYSNAME = MPZ3
SMF123S2_MVS_ASID = 54
SMF123S2_MVS_SID = MPZ3
SMF123S2_REQ_ID = 732
SMF123S2_TRACKING_TOKEN = 0x42415131 77734859 41514159 314E6670 31395046
35304455 312B6E7A 51454241
514E6F76 77583159 7275414F 40404040 40404040 40404040 40404040
SMF123S2_REQ_HDR1 =
SMF123S2_REQ_HDR2 =
SMF123S2_REQ_HDR3 =
```

CICS Performance Analyzer

V5R4M0		CICS Performance Analyzer z/OS Connect Summary					
ZCEE0001 Printed at 13:35:01 8/21/2021		Data from 11:30:24 8/19/2021 to 12:11:24 8/19/2021				Page 1	
Initial CICS PA report							
JOBNAME : BAQSTRT SPNAME : imsmobile-2.0							
Request: 49 Fail: 0 Timed out: 0 Get: 49 Post: 0 Put: 0 Delete: 0							
----- Maximum value Request details -----							
SOR Sent Latency	Avg .0326	Max .3781	Req ID 551	ZC Entry 19/08/2021 12:09:45.036778			
SOR Response	.0016	.0183	551	19/08/2021 12:09:45.036778			
ZC Exit Latency	.0025	.0048	504	19/08/2021 12:09:36.823661			
ZC Response	.0367	.3982	551	19/08/2021 12:09:45.036778			
ZC Time	.0351	.3799	551	19/08/2021 12:09:45.036778			
JOBNAME : BAQSTRT SPNAME : restclient-1.0							
Request: 50 Fail: 50 Timed out: 0 Get: 50 Post: 0 Put: 0 Delete: 0							
----- Maximum value Request details -----							
SOR Sent Latency	Avg .0478	Max .5953	Req ID 488	ZC Entry 19/08/2021 12:09:33.386614			
SOR Response	.0027	.0127	594	19/08/2021 12:09:52.016624			
ZC Exit Latency	.0014	.0029	524	19/08/2021 12:09:40.369997			
ZC Response	.0519	.6004	488	19/08/2021 12:09:33.386614			
ZC Time	.0492	.5972	488	19/08/2021 12:09:33.386614			
JOBNAME : BAQSTRT SPNAME : CICS-1.0							
Request: 49 Fail: 0 Timed out: 0 Get: 49 Post: 0 Put: 0 Delete: 0							
----- Maximum value Request details -----							
SOR Sent Latency	Avg .0300	Max .0589	Req ID 450	ZC Entry 19/08/2021 12:09:26.478282			
SOR Response	.0011	.0049	517	19/08/2021 12:09:39.019456			
ZC Exit Latency	.0077	.0138	450	19/08/2021 12:09:26.478282			
ZC Response	.0387	.0741	450	19/08/2021 12:09:26.478282			
ZC Time	.0376	.0727	450	19/08/2021 12:09:26.478282			

IBM z Omegamon for JVM

The image displays three windows from the IBM z Omegamon for JVM interface:

- WG31 - 3270**: A main window titled "z/OS Connect Request Summary". It shows a table of requests with columns: APIName, Service, SoR ID, Reference, Resource. The table includes rows for "Last 30 Minute(s)", "Last 1 Hour(s)", and "Date/Time Range". Below the table are sections for "Columns" and "API Name" (listing services like catalog, db2employee, filea, filequeue, *ADMIN*, phonebook).
- WG31 - 3270**: A window titled "Requests by Service Name". It shows a table with columns: Service, Request, Error, Timeout, Response Time, and OS Conn Avg. The table lists services like inquireSingle, cscvincService, and selectEmployee.
- WG31 - 3270**: A window titled "z/OS Connect Request Detail". It displays a detailed log of a specific request with various parameters and their values.

IBM z Omegamon for JVM

WG31 - 3270

File Edit View Communication Actions Window Help

File Edit View Tools Navigate Help 04/02/2019 18:59:29
Auto Update : Off
SMF ID : WG31
Coll ID : KJJ1

Command ==> KJJZCDD z/OS Connect Request Detail

```

Event time..... 04/02/19 18:49:14.525
Request Type... API
API name.... filequeue
Request URI... /filequeue/mq
Query String...
Method..... GET
Port..... 9453
HTTP code.... 200 (OK)
Timeout.... No
Service Name.. FileaQueue
Total Req Time. 0.016206s
z/OS Conn Time. 0.016206s
SoR Resp Time. 0.000000s
SoR ID.... NONE
SoR Ref.... NONE
SoR Resource. NONE
Remote Address. 192.168.0.141
Request Length. 0
Response Length. 191
Correlator.... e6e2d3d7d3c5e7400011000010d5ea51
Operation.... getFilea
Provider.... IBM MQ for z/OS
User ID.... Fred

```

VERIFY | BACK | HOME | Hub WG31:CMS on platform WG31(z/OS) 01/002

Connected to remote server/host wg31a using lu/pool TCP00109 and port 23

Event time..... 04/02/19 18:48:34.790
Request Type... API
API name.... db2employee
Request URI... /db2/employee/000020
Query String...
Method..... GET
Port..... 9453
HTTP code.... 200 (OK)
Timeout.... No
Service Name.. selectEmployee
Total Req Time. 0.022592s
z/OS Conn Time. 0.022592s
SoR Resp Time. 0.000000s
SoR ID.... NONE
SoR Ref.... NONE
SoR Resource. NONE
Remote Address. 192.168.0.141
Request Length. 0
Response Length. 326
Correlator.... e6e2d3d7d3c5e7400011000010d5ea50
Operation.... getSelectEmployee
Provider.... restclient-1.0
User ID.... Fred

VERIFY | BACK | HOME | Hub WG31:CMS on platform WG31(z/OS) 01/002

Connected to remote server/host wg31a using lu/pool TCP00109 and port 23

WG31 - 3270

File Edit View Communication Actions Window Help

File Edit View Tools Navigate Help 04/02/2019 19:00:52
Auto Update : Off
SMF ID : WG31
Coll ID : KJJ1

Command ==> KJJZCDD z/OS Connect Request Detail

```

Event time..... 04/02/19 18:47:54.267
Request Type... API
API name.... cscvinc
Request URI... /cscvinc/employee/444444
Query String...
Method..... GET
Port..... 9453
HTTP code.... 200 (OK)
Timeout.... No
Service Name.. cscvincService
Total Req Time. 0.008006s
z/OS Conn Time. 0.005515s
SoR Resp Time. 0.002491s
SoR ID.... USIBMWZ .CICS59Z
SoR Ref.... cscvinc
SoR Resource. CSMI_CSCVINC
Remote Address. 192.168.0.141
Request Length. 0
Response Length. 302
Correlator.... e6e2d3d7d3c5e7400011000010d5ea50
Operation.... getCscvincService
Provider.... CICS-1.0
User ID.... Fred

```

VERIFY | BACK | HOME | Hub WG31:CMS on platform WG31(z/OS) 01/002

Connected to remote server/host wg31a using lu/pool TCP00109 and port 23

Event time..... 04/02/19 19:07:04.090
Request Type... API
API name.... phonebook
Request URI... /phonebook/contacts/LAST1
Query String...
Method..... GET
Port..... 9453
HTTP code.... 200 (OK)
Timeout.... No
Service Name.. ivtnoService
Total Req Time. 0.345265s
z/OS Conn Time. 0.169460s
SoR Resp Time. 0.181805s
SoR ID.... IVPN
SoR Ref.... IVTNO
SoR Resource. IVTNO
Remote Address. 192.168.0.141
Request Length. 0
Response Length. 158
Correlator.... e6e2d3d7d3c5e7400011000010d5ea55
Operation.... getPhoneBookService1
Provider.... imsmobile-2.0
User ID.... Fred

VERIFY | BACK | HOME | Hub WG31:CMS on platform WG31(z/OS) 01/002

Connected to remote server/host wg31a using lu/pool TCP00109 and port 23