



# SSL Guide

---

Version: 2022.1.0

# Copyright AppViewX, Inc.

**Copyright © 2020 AppViewX, Inc. All Rights Reserved.**

This document may not be copied, disclosed, transferred, or modified without the prior written consent of AppViewX, Inc. While all content is believed to be correct at the time of publication, it is provided as general-purpose information. The content is subject to change without notice and is provided “as is” and with no expressed or implied warranties whatsoever, including, but not limited to, a warranty for accuracy made by AppViewX. The software described in this document is provided under written license only, contains valuable trade secrets and proprietary information, and is protected by the copyright laws of the United States and other countries. Unauthorized use of software or its documentation can result in civil damages and criminal prosecution.

## **Trademarks**

The trademarks, logos, and service marks displayed in this manual are the property of AppViewX or other third parties. Users are not permitted to use these marks without the prior written consent of AppViewX or such third party which may own the mark.

## **External Reference Links**

This product includes software developed by the CentOS Project ([www.centos.org](http://www.centos.org)).

This product includes software developed by Red Hat, Inc. ([www.redhat.com](http://www.redhat.com)).

This product includes software developed by VMware, Inc. ([www.vmware.com](http://www.vmware.com)).

All other trademarks mentioned in this document are the property of their respective owners.

## **Contact Information**

AppViewX, Inc.

222 Broadway, FL 19

New York, NY 10038

Email: [info@appviewx.com](mailto:info@appviewx.com)

Web: [www.appviewx.com](http://www.appviewx.com)

# Contents

Copyright AppViewX, Inc.....	ii
Copyright © 2020 AppViewX, Inc. All Rights Reserved.....	ii
Trademarks.....	ii
External Reference Links.....	ii
Contact Information.....	ii
Preface.....	iv
Revision History.....	iv
About this Guide .....	iv
Audience.....	iv
Text Conventions.....	iv
<b>Chapter 1. Setting Up Function Groups and Web Services.....</b>	<b>5</b>
<b>Chapter 2. More Information.....</b>	<b>38</b>
Documentation Feedback.....	38
Requesting Technical Support.....	38
Self-Help Online Tools and Resources.....	38

# Preface

## Revision History

Revision	Description	Date
1.0	SSL Guide V3	08 Nov 2022

## About this Guide

This document explains the process of setting up function groups and web services in the customer's SAP ABAP environment.

## Audience

This guide is intended for the customers of AppViewX using the SAP ABAP environment.

## Text Conventions

The following text conventions are used in this document:

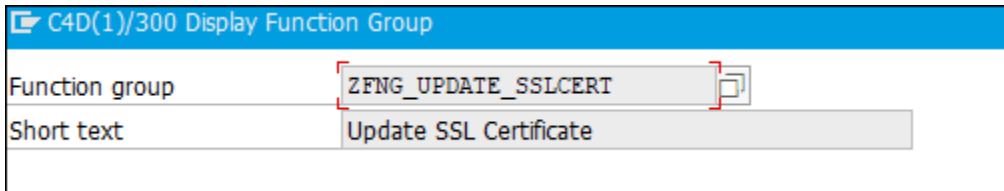
Convention	Description
<b>boldface</b>	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
<i>italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
<code>codeblock</code>	Indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

# Chapter 1: Setting Up Function Groups and Web Services

This section explains the process of setting up function groups and web service in the customer's SAP ABAP environment.

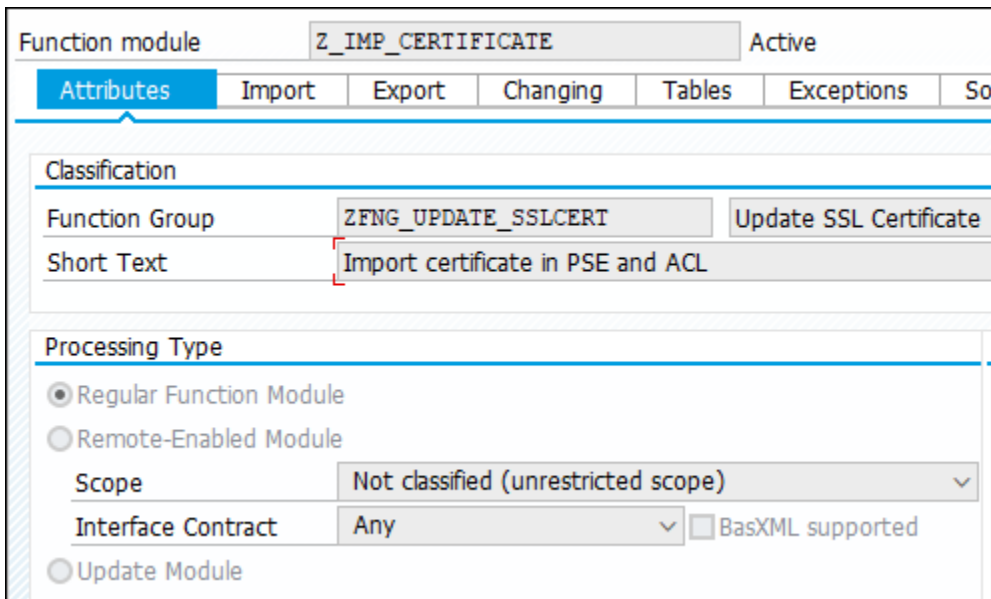
The process involves creating a *Function Group*, *Service Definition*, and a *Binding in SOA*.

1. Create a **Function Group** as shown below.



The screenshot shows the 'C4D(1)/300 Display Function Group' dialog box. It contains two input fields: 'Function group' with the value 'ZFNG\_UPDATE\_SSLCERT' and 'Short text' with the value 'Update SSL Certificate'. Red dashed boxes highlight the text in both fields.

2. Create a function module **Z\_IMP\_CERTIFICATE** as shown below.



The screenshot shows the configuration for the function module 'Z\_IMP\_CERTIFICATE'. The 'Attributes' tab is active. Under the 'Classification' section, the 'Function Group' is set to 'ZFNG\_UPDATE\_SSLCERT' and the 'Short Text' is 'Import certificate in PSE and ACL'. Under the 'Processing Type' section, 'Regular Function Module' is selected, 'Scope' is 'Not classified (unrestricted scope)', and 'Interface Contract' is 'Any'. There is also a checkbox for 'BasXML supported' which is currently unchecked.

Function module **Z\_IMP\_CERTIFICATE** Active

Attributes **Import** Export Changing Tables Exceptions Source code

Parameter Name	Typi...	Associated Type	Default value	Op...	Pa...	Short text
I_CERTIFICATE_XCERT	TYPE	XSTRING		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
I_CERTIFICATE_IMPORT...	TYPE	STRING		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
I_CERTIFICATE_IMPORT...	TYPE	STRING		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
I_EP_SID	TYPE	WPS_SYSID		<input type="checkbox"/>	<input checked="" type="checkbox"/>	R/3 System, name of R/3 System
I_EP_CLIENT	TYPE	WPS_MANDT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	SAP System, Client Number from Log...
I_PSENAME	TYPE	SSFPSENAME		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	PSE file name
I_INSTANCENO	TYPE	NUMC2		<input type="checkbox"/>	<input checked="" type="checkbox"/>	Instance No.

Function module **Z\_IMP\_CERTIFICATE** Active

Attributes Import **Export** Changing Tables Exceptions Source code

Parameter Name	Typing	Associated Type	Pass by ...	Short text
RETURN	TYPE	BAPIRET2	<input checked="" type="checkbox"/>	Return Parameter(s)

Function module <b>Z_IMP_CERTIFICATE</b> Active	
Attributes Import Export Changing Tables Exceptions Source	
Classes	
Exception	Short text
NO_AUTHORIZATION	No authorization for transaction STRUSTSSO2
FILE_NOT_EXIST	File doesn't exist
GET_SAPSYS_PSE_FAILED	
ENQUEUE_PSE_FAILED	
LOAD_PSE_FAILED	
UPLOAD_FILE_FAILED	
PARSE_CERTIFICATE_FAILED	
REMOVE_CERTIFICATE_FAILED	
PUT_CERTIFICATE_FAILED	
STORE_PSE_FAILED	
DEQUEUE_PSE_FAILED	
OPEN_TEMP_PSE_FAILED	
DELETE_TEMP_PSE_FAILED	
ADD_CERTIFICATE_TO_ACL_FAILED	

3. Copy the source code below for function module **Z\_IMP\_CERTIFICATE**.

```

FUNCTION z_imp_certificate.
*-----
***Local Interface:
** IMPORTING
**  VALUE(I_CERTIFICATE_XCERT) TYPE XSTRING OPTIONAL
**  VALUE(I_CERTIFICATE_IMPORT_PATH) TYPE STRING OPTIONAL
**  VALUE(I_CERTIFICATE_IMPORT_FILE) TYPE STRING OPTIONAL
**  VALUE(I_EP_SID) TYPE WPS_SYSID
**  VALUE(I_EP_CLIENT) TYPE WPS_MANDT
**  VALUE(I_PSENAME) TYPE SSFPSENAME OPTIONAL
**  VALUE(I_INSTANCENO) TYPE NUMC2
** EXPORTING
**  VALUE(RETURN) TYPE BAPIRET2
** EXCEPTIONS
**  NO_AUTHORIZATION
**  FILE_NOT_EXIST

```

```

** GET_SAPSYS_PSE_FAILED
** ENQUEUE_PSE_FAILED
** LOAD_PSE_FAILED
** UPLOAD_FILE_FAILED
** PARSE_CERTIFICATE_FAILED
** REMOVE_CERTIFICATE_FAILED
** PUT_CERTIFICATE_FAILED
** STORE_PSE_FAILED
** DEQUEUE_PSE_FAILED
** OPEN_TEMP_PSE_FAILED
** DELETE_TEMP_PSE_FAILED
** ADD_CERTIFICATE_TO_ACL_FAILED
**-----

DATA:
  l_certificate_xcert    TYPE xstring,
  l_certificate_import_path TYPE string,
  l_certificate_import_file TYPE string,
  l_subject             TYPE string,
  l_issuer              TYPE string,
  l_snumber             TYPE string,
  l_filename            TYPE string,
  l_result              TYPE abap_bool,
  l_cert               TYPE string,
  l_length              TYPE i,
  l_offset              TYPE i,
  l_rc                  TYPE i,      "err: 1=get own cert, 2=get certlist
  l_message             TYPE string,
  l_certatt             TYPE certattr,
  l_profile             TYPE ssfargs-profile,
  l_psename             TYPE ssf_pse_h-filename, " VALUE 'SAPSYS.pse',
  l_fname              TYPE rlgrap-filename,
  l_fprofile            TYPE ssfargs-profile,
  l_s_certlistattr     TYPE certattr,
  l_s_twpsso2acl       TYPE twpsso2acl,
  l_separator          TYPE char1.

```

```
AUTHORITY-CHECK OBJECT 'S_TCODE'  
    ID 'TCD' FIELD 'STRUSTSSO2'.  
  
IF sy-subrc <> 0.  
    return-type = 'A'.  
    return-number = '1'.  
    return-message = 'no_authorization'.  
    EXIT.  
ENDIF.  
  
IF i_psename IS INITIAL.  
    l_psename = 'SAPSYS.pse'.  
ELSE.  
    l_psename = i_psename.  
ENDIF.  
  
l_certificate_xcert = i_certificate_xcert.  
  
IF l_certificate_xcert IS INITIAL.  
  
    CALL METHOD cl_gui_frontend_services=>get_file_separator  
    CHANGING  
        file_separator    = l_separator  
    EXCEPTIONS  
        cntl_error       = 1  
        error_no_gui     = 2  
        not_supported_by_gui = 3  
        OTHERS           = 4.  
  
    IF sy-subrc <> 0.  
        "assume windows  
        l_separator = '\\'.  
    ENDIF.  
  
    l_certificate_import_path = i_certificate_import_path.  
    l_certificate_import_file = i_certificate_import_file.  
  
    l_length = strlen( l_certificate_import_path ) - 1.
```

```

IF l_length >= 0.

  IF l_certificate_import_path+l_length(1) <> l_separator.

    CONCATENATE l_certificate_import_path l_separator INTO l_certificate_import_path.

  ENDIF.

ENDIF.

IF strlen( l_certificate_import_file ) > 0.

  IF l_certificate_import_file(1) = l_separator.

    l_certificate_import_file = l_certificate_import_file+1.

  ENDIF.

ENDIF.

CONCATENATE l_certificate_import_path l_certificate_import_file INTO l_filename.

IF l_filename CS '<BW_SID>'.

  return-type = 'A'.

  return-number = '2'.

  return-message = 'file_not_exist'.

  EXIT.

ENDIF.

CALL METHOD cl_gui_frontend_services=>file_exist

EXPORTING

  file = l_filename

RECEIVING

  result = l_result.

IF l_result IS INITIAL.

  return-type = 'A'.

  return-number = '2'.

  return-message = 'file_not_exist'.

  EXIT.

ENDIF.

ENDIF.

CALL FUNCTION 'SSF_GET_PARAMETER'

EXPORTING

  mandt          = sy-mandt

  application    = 'SSO2'

```

```

* application      = 'DEFAULT'

IMPORTING

str_profile       = l_profile

EXCEPTIONS

ssf_parameter_not_found = 1

OTHERS           = 2.

IF sy-subrc <> 0.

return-type = 'A'.

return-number = '3'.

return-message = 'get_sapsys_pse_failed'.

EXIT.

ENDIF.

CALL FUNCTION 'SSFPSE_ENQUEUE'

EXPORTING

psename          = l_psename

* HOST           =

instanceid       = i_instanceno '00'

EXCEPTIONS

database_failed = 1

foreign_lock    = 2

internal_error  = 3

OTHERS          = 4.

IF sy-subrc <> 0.

return-type = 'A'.

return-number = '4'.

return-message = 'enqueue_pse_failed'.

EXIT.

ENDIF.

CALL FUNCTION 'SSFPSE_LOAD'

EXPORTING

psename          = l_psename

* HOST           =

instanceid       = i_instanceno '00'

IMPORTING

* PSELEN         =

```

```

* ID      =
  fname   = I_fname
* B_FALLBACK =
* TABLES
* PSE     = I_t_pse
EXCEPTIONS
  authority_missing = 1
  database_failed  = 2
  file_write_failed = 3
  OTHERS          = 4.
IF sy-subrc <> 0.
  return-type = 'A'.
  return-number = '5'.
  return-message = 'load_pse_failed'.
EXIT.
ENDIF.

IF I_certificate_xcert IS INITIAL.
  PERFORM readws_binary_item
    USING I_filename
    CHANGING I_certificate_xcert
      I_rc.
  IF I_rc = 1.
    return-type = 'A'.
    return-number = '6'.
    return-message = 'upload_file_failed'.
    EXIT.
  ENDIF.
ENDIF.

I_fprofile = I_fname.
CALL FUNCTION 'SSFC_PUT_CERTIFICATE'
  EXPORTING
    profile = I_fprofile
* PROFILEPW = ''
  certificate = I_certificate_xcert
  add_to_cab = abap_true "newly added

```

```

EXCEPTIONS

ssf_krn_error      = 1

ssf_krn_nomemory   = 2

ssf_krn_nosslib    = 3

ssf_krn_invalid_par = 4

ssf_krn_certexists = 5

OTHERS             = 6.

IF sy-subrc <> 0.

* if certificate already exists -> Delete it and upload it once again

IF sy-subrc = 5.

* Determine issuer and subject from certificate

CALL FUNCTION 'SSFC_PARSE_CERTIFICATE'

EXPORTING

  certificate      = l_certificate_xcert

IMPORTING

  subject         = l_subject

  issuer          = l_issuer

  serialno       = l_snumber

EXCEPTIONS

ssf_krn_error      = 1

ssf_krn_nomemory   = 2

ssf_krn_nosslib    = 3

ssf_krn_invalid_par = 4

OTHERS             = 5.

IF sy-subrc <> 0.

  return-type = 'A'.

  return-number = '7'.

  return-message = 'parse_certificate_failed'.

EXIT.

ENDIF.

* remove certificate

CALL FUNCTION 'SSFC_REMOVECERTIFICATE'

EXPORTING

  profile         = l_fprofile

```

```

subject      = l_subject
issuer       = l_issuer
serialno     = l_snumber

EXCEPTIONS

ssf_krn_error      = 1

ssf_krn_nomemory   = 2

ssf_krn_nossflib   = 3

ssf_krn_invalid_par = 4

ssf_krn_nocertificate = 5

OTHERS            = 6.

IF sy-subrc <> 0.

return-type = 'A'.

return-number = '8'.

return-message = 'remove_certificate_failed'.

EXIT.

ENDIF.

* upload certificate once again (new certificate)

CALL FUNCTION 'SSFC_PUT_CERTIFICATE'

EXPORTING

profile      = l_fprofile

* PROFILEPW   = ""

certificate  = l_certificate_xcert

EXCEPTIONS

ssf_krn_error      = 1

ssf_krn_nomemory   = 2

ssf_krn_nossflib   = 3

ssf_krn_invalid_par = 4

ssf_krn_certexists = 5

OTHERS            = 6.

IF sy-subrc <> 0.

return-type = 'A'.

return-number = '9'.

return-message = 'put_certificate_failed'.

EXIT.

```

```

ENDIF.

ELSE.

return-type = 'A'.

return-number = '9'.

return-message = 'put_certificate_failed'.

EXIT.

ENDIF.

ENDIF.

CALL FUNCTION 'SSFPSE_STORE'

EXPORTING

  fname      = l_fname

* PSEPIN     =

  purname    = l_purname

* ID        =

* HOST      = ''

  instanceid = i_instanceno '00'

* TYPE      = 'PSE'

* FORMAT    = 'RAW'

* B_NEWDN   =

* B_CLEANUP = 'X'

  b_distribute = 'X'

EXCEPTIONS

  file_load_failed = 1

  storing_failed   = 2

  authority_missing = 3

  OTHERS          = 4.

IF sy-subrc <> 0.

return-type = 'A'.

return-number = '10'.

return-message = 'store_pse_failed'.

EXIT.

ENDIF.

CALL FUNCTION 'SSFPSE_DEQUEUE'

EXPORTING

```

```

    psetname      = l_psetname
*   HOST          =
    instanceid    = i_instanceno" '00'

EXCEPTIONS

    database_failed = 1

    foreign_lock    = 2

    internal_error  = 3

    OTHERS          = 4.

IF sy-subrc <> 0.

    return-type = 'A'.

    return-number = '11'.

    return-message = 'dequeue_pse_failed'.

EXIT.

ENDIF.

TRY.

    DELETE DATASET l_fname.

CATCH cx_sy_file_open.

    return-type = 'A'.

    return-number = '12'.

    return-message = 'delete_temp_pse_failed'.

EXIT.

CATCH cx_sy_file_authority.

    return-type = 'A'.

    return-number = '13'.

    return-message = 'delete_temp_pse_failed'.

EXIT.

ENDTRY.

* Add certificate to ACL

CALL FUNCTION 'SSFP_PARSECERTIFICATE'

EXPORTING

    certxstring    = l_certificate_xcert

IMPORTING

    certattributes = l_s_certlistattr

EXCEPTIONS

    ssf_parsing_error = 1

```

```

OTHERS      = 2.

IF sy-subrc EQ 0.

  l_s_twpsso2acl-wps_sysid = i_ep_sid.

  l_s_twpsso2acl-wps_mandt = i_ep_client.

  l_s_twpsso2acl-dn_subject = l_s_certlistattr-subject.

  l_s_twpsso2acl-dn_issuer = l_s_certlistattr-issuer.

  l_s_twpsso2acl-serialno = l_s_certlistattr-snumber.

  MODIFY twpsso2acl FROM l_s_twpsso2acl.

  IF sy-subrc <> 0.

    return-type = 'A'.

    return-number = '14'.

    return-message = 'add_certificate_to_acl_failed'.

  EXIT.

ENDIF.

ENDIF.

ENDFUNCTION.

*&-----*
*&  Form READWS_BINARY_ITEM
*&-----*
*   text
*-----*
* -->FILENAME text
* -->ITEMBIN text
* -->ITEMLENG text
* -->RC text
*-----*

FORM readws_binary_item

  USING filename TYPE csequence

  CHANGING itembin TYPE xsequence

  rc TYPE i. "errors: 1=file open, 2=empty, 3=toolong

DATA: itemtype TYPE c.

DATA: maxleng TYPE i VALUE 2147483647. "maxint

DATA: s_filename TYPE string.

DATA: line(1024) TYPE x.

DATA: xtab(1024) TYPE x OCCURS 0.

DATA: itemleng TYPE i.

```

```

CLEAR rc.

DESCRIBE FIELD itembin TYPE itemtype.

IF itemtype = 'X'.

    DESCRIBE FIELD itembin LENGTH maxleng IN BYTE MODE.

ENDIF.

s_filename = filename.

CALL FUNCTION 'GUI_UPLOAD'

EXPORTING

    filename      = s_filename
    filetype      = 'BIN'
*   HAS_FIELD_SEPARATOR = ''
*   HEADER_LENGTH   = 0

IMPORTING

    filelength    = itemleng
*   HEADER         =

TABLES

    data_tab      = xtab

EXCEPTIONS

    file_open_error   = 1
    file_read_error   = 2
    no_batch          = 3
    gui_refuse_filetransfer = 4
    invalid_type      = 5
    no_authority      = 6
    unknown_error     = 7
    bad_data_format   = 8
    header_not_allowed = 9
    separator_not_allowed = 10
    header_too_long   = 11
    unknown_dp_error  = 12
    access_denied     = 13
    dp_out_of_memory  = 14
    disk_full         = 15
    dp_timeout        = 16
    OTHERS           = 17.

```

```

IF sy-subrc NE 0.
    rc = 1. RETURN.
ENDIF.

IF itmleng EQ 0.    "item not found
    rc = 2. RETURN.
ELSEIF itmleng > maxleng. "item too long
    rc = 3. RETURN.
ELSE.
    CLEAR itembin.
    LOOP AT xtab INTO line.
        CONCATENATE itembin line INTO itembin IN BYTE MODE.
    ENDLOOP.
    itembin = itembin(itmleng).
ENDIF.

ENDFORM.          "READWS_BINARY
    
```

4. Create a remote enabled function module **Z\_UPD\_SSL\_CERT** as shown below.

Function module		Z_UPD_SSL_CERT	Active							
<table border="1"> <tr> <td>Attributes</td> <td>Import</td> <td>Export</td> <td>Changing</td> <td>Tables</td> <td>Exceptions</td> <td>Source code</td> </tr> </table>				Attributes	Import	Export	Changing	Tables	Exceptions	Source code
Attributes	Import	Export	Changing	Tables	Exceptions	Source code				
Classification										
Function Group	ZFNG_UPDATE_SSLCERT	Update SSL Certificate								
Short Text	Update SSL Certificate									
Processing Type			General Data							
<input type="radio"/> Regular Function Module <input checked="" type="radio"/> Remote-Enabled Module			Person Responsible							
Scope	Not classified (unrestricted scope)		Last Changed By							
Interface Contract	Any	<input type="checkbox"/> BasXML supported	Changed on							
			Package							

Function module **Z\_UPD\_SSL\_CERT** Active

Attributes **Import** Export Changing Tables Exceptions Source code

Parameter Name	Typi...	Associated Type	Default value	Op...	Pa...	Short text	Lo...
IM_CERT	TYPE	STRING		<input type="checkbox"/>	<input checked="" type="checkbox"/>	Path	<input checked="" type="checkbox"/> D.
IM_PSENAME	TYPE	SSFPSENAME		<input type="checkbox"/>	<input checked="" type="checkbox"/>	PSE file name	<input checked="" type="checkbox"/> D.
IM_HOST	TYPE	SYHOST		<input type="checkbox"/>	<input checked="" type="checkbox"/>	Application Server	<input checked="" type="checkbox"/> D.
IM_INSTANCENO	TYPE	NUMC2		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Instance No.	<input checked="" type="checkbox"/> D.
IM_OWNCERT	TYPE	CHAR01		<input type="checkbox"/>	<input checked="" type="checkbox"/>	Own Certificate	<input checked="" type="checkbox"/> D.
				<input type="checkbox"/>	<input type="checkbox"/>		

Function module **Z\_UPD\_SSL\_CERT** Active

Attributes Import **Export** Changing Tables Exceptions Source code

Parameter Name	Typing	Associated Type	Pass by ...	Short text
EX_ERROR	TYPE	CHAR01	<input checked="" type="checkbox"/>	Error Flag
EX_TEXT	TYPE	CHAR100	<input checked="" type="checkbox"/>	Text

5. Copy the source code below for function module **Z\_UPD\_SSL\_CERT**.

```

FUNCTION z_upd_ssl_cert.
**-----
**Local Interface:
** IMPORTING
**   VALUE(IM_CERT) TYPE STRING
**   VALUE(IM_PSENAME) TYPE SSFPSENAME
**   VALUE(IM_HOST) TYPE SYHOST
**   VALUE(IM_INSTANCENO) TYPE NUMC2 OPTIONAL
**   VALUE(IM_OWNCERT) TYPE CHAR01
** EXPORTING
**   VALUE(EX_ERROR) TYPE CHAR01
**   VALUE(EX_TEXT) TYPE CHAR100
**-----
ex_error = 'X'.

IF im_instanceno IS INITIAL.
  im_instanceno = '00'.
ENDIF.

```

```
IF im_owncert NE 'X'.
```

```
DATA: ld_xstr TYPE xstring.
```

```
CALL FUNCTION 'SCMS_STRING_TO_XSTRING'
```

```
EXPORTING
```

```
text = im_cert
```

```
IMPORTING
```

```
buffer = ld_xstr
```

```
EXCEPTIONS
```

```
failed = 1
```

```
OTHERS = 2.
```

```
CALL FUNCTION 'Z_IMP_CERTIFICATE'
```

```
EXPORTING
```

```
i_certificate_xcert = ld_xstr
```

```
i_ep_sid = sy-sysid " R/3 System, name of R/3 System
```

```
i_ep_client = sy-mandt " SAP System, Client Number from Logon
```

```
i_psename = im_psename "SAPSSLA.pse'
```

```
i_instanceno = im_instanceno
```

```
EXCEPTIONS
```

```
no_authorization = 1 " No authorization for transaction STRUSTSSO2
```

```
file_not_exist = 2 " File doesn't exist
```

```
get_sapsys_pse_failed = 3
```

```
enqueue_pse_failed = 4
```

```
load_pse_failed = 5
```

```
upload_file_failed = 6
```

```
parse_certificate_failed = 7
```

```
remove_certificate_failed = 8
```

```
put_certificate_failed = 9
```

```
store_pse_failed = 10
```

```
dequeue_pse_failed = 11
```

```
open_temp_pse_failed = 12
```

```
delete_temp_pse_failed = 13
```

```
add_certificate_to_acl_failed = 14
```

```
OTHERS          = 15.  
  
IF sy-subrc EQ 0.  
  CLEAR: ex_error.  
  
ELSE.  
  CASE sy-subrc.  
    WHEN '1'.  
      ex_text = 'No authorization'.  
    WHEN '2'.  
      ex_text = 'File does not exist'.  
    WHEN '3'.  
      ex_text = 'Get sapsys PSE failed'.  
    WHEN '4'.  
      ex_text = 'Enqueue PSE failed'.  
    WHEN '5'.  
      ex_text = 'Load PSE failed'.  
    WHEN '6'.  
      ex_text = 'Upload file failed'.  
    WHEN '7'.  
      ex_text = 'Parse certificate failed'.  
    WHEN '8'.  
      ex_text = 'Remove certificate failed'.  
    WHEN '9'.  
      ex_text = 'Put certificate failed'.  
    WHEN '10'.  
      ex_text = 'Store PSE failed'.  
    WHEN '11'.  
      ex_text = 'Dequeue PSE failed'.  
    WHEN '12'.  
      ex_text = 'Open temp PSE failed'.  
    WHEN '13'.  
      ex_text = 'Delete temp PSE failed'.  
    WHEN '14'.  
      ex_text = 'Add certificate to acl failed'.  
    WHEN '15'.  
      ex_text = 'Other issue'.  
  ENDCASE.
```

```

ENDIF.
ELSE: "OWN Certificate
DATA: lt_certresp TYPE ssfcertreq,
      lw_certresp LIKE LINE OF lt_certresp,
      lt_ret      TYPE TABLE OF bapiret2.

DATA(ld_len) = strlen( im_cert ).

DATA: ld_count TYPE i,
      ld_begin TYPE i,
      ld_end   TYPE i,
      ld_length TYPE i.

cl_abap_pse=>convert_string_to_certreqtab(
EXPORTING
  iv_string   = im_cert
IMPORTING
  et_certreqtab = DATA(lt_certtab)
).
LOOP AT lt_certtab INTO DATA(lw_certtab).
  lw_certresp = lw_certtab-line.
  APPEND lw_certresp TO lt_certresp.
ENDLOOP.

CALL FUNCTION 'Z_IMP_CERTIFICATERESPONSE'
EXPORTING
  it_certresponse = lt_certresp
  iv_certresponse_len = ld_len
  i_psename       = im_psename
  i_instanceno    = im_instanceno
  i_host          = im_host
TABLES
  et_bapiret2     = lt_ret.
CLEAR: ex_error.

READ TABLE lt_ret WITH KEY type = 'E' TRANSPORTING NO FIELDS.
IF sy-subrc EQ 0.

```

```

ex_error = 'X'.

ex_text = 'Error updating own certificate'.

ENDIF.

ENDIF.

ENDFUNCTION.
    
```

6. Create a Remote Enabled Function Module **Z\_IMP\_CERTIFICATERESPONSE** as shown below.

Function module **Z\_IMP\_CERTIFICATERESPONSE** Active

Attributes Import Export Changing Tables Exceptions Sou

**Classification**

Function Group **ZFNG\_UPDATE\_SSLCERT** Update SSL Certificate

Short Text **Import Certificate Response**

**Processing Type**

Regular Function Module

Remote-Enabled Module

Scope **Not classified (unrestricted scope)**

Interface Contract **Any**  BasXML supported

Update Module

Function module **Z\_IMP\_CERTIFICATERESPONSE** Active

Attributes **Import** Export Changing Tables Exceptions Source code

Parameter Name	Typi...	Associated Type	Default value	Op...	Pa...	Short text
IT_CERTRESPONSE	TYPE	SSFCERTREQ		<input type="checkbox"/>	<input checked="" type="checkbox"/>	
IV_CERTRESPONSE_LEN	TYPE	SSFLEN		<input type="checkbox"/>	<input checked="" type="checkbox"/>	
IV_PSEPIN	TYPE	SSFPABPW		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
I_PSENAME	TYPE	SSFPSENAME		<input type="checkbox"/>	<input checked="" type="checkbox"/>	PSE file name
I_INSTANCENO	TYPE	NUMC2		<input type="checkbox"/>	<input checked="" type="checkbox"/>	Instance No.
I_HOST	TYPE	SYHOST		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Hostname

Function module		Z_IMP_CERTIFICATERESPONSE		Active	
Attributes		Import		Export	
Changing		Tables		Exceptions	
Source code					
Parameter Name	Typing	Associated Type	Optional	Short text	
ET_BAPIRET2	LIKE	BAPIRET2	<input type="checkbox"/>		

7. Copy the source code below for the remote enabled function module

### Z\_IMP\_CERTIFICATERESPONSE.

```

FUNCTION z_imp_certificateresponse.
**-----
****Local Interface:
** IMPORTING
**  VALUE(IT_CERTRESPONSE) TYPE SSFCERTREQ
**  VALUE(IV_CERTRESPONSE_LEN) TYPE SSFLEN
**  VALUE(IV_PSEPIN) TYPE SSFPABPW OPTIONAL
**  VALUE(I_PSENAME) TYPE SSFPSENAME
**  VALUE(I_INSTANCENO) TYPE NUMC2
**  VALUE(I_HOST) TYPE SYHOST OPTIONAL
** TABLES
**  ET_BAPIRET2 STRUCTURE BAPIRET2
**-----

DATA:
  lv_psename TYPE ssfpsename,
  lv_fname  TYPE localfile,
  lv_pab    TYPE ssfpab,
  lv_id     TYPE ssf_pse_h-id.

lv_psename = i_psename.
*-----
* Download PSE from database
*-----

CALL FUNCTION 'SSFPSE_LOAD'

EXPORTING
  psename      = lv_psename

```

```

* host      = i_host

* instanceid = i_instanceno

IMPORTING

* pselen    =

id          = ld_id

fname      = lv_fname

EXCEPTIONS

authority_missing = 1

database_failed = 2

file_write_failed = 3

OTHERS      = 4.

* -----

IF sy-subrc <> 0.

RETURN.

ENDIF.

* -----

* Put certificate response to PSE

* -----

lv_pab = lv_fname.

CALL FUNCTION 'SSFC_PUTCERTIFICATERESPONSE'

EXPORTING

profile      = lv_pab

profilepw    = iv_psepin

certresponse_len = iv_certresponse_len

TABLES

certresponse = it_certresponse

EXCEPTIONS

ssf_krn_error = 1

ssf_krn_nomemory = 2

ssf_krn_nosflib = 3

ssf_krn_invalid_par = 4

ssf_krn_invalidcertresponse = 5

OTHERS      = 6.

* -----

IF sy-subrc <> 0.

DELETE DATASET lv_fname. "temporary PSE file (SSFPSE_LOAD)

```

```

RETURN.
ENDIF.

* -----
* Store PSE to database
* -----

CALL FUNCTION 'SSFPSE_STORE'

EXPORTING

  fname      = lv_fname
*  psepin    = iv_psepin
  psename    = lv_psename
  id         = ld_id
*  host      = i_host
*  instanceid = i_instanceno
  type       = 'LPSE'
  format     = 'RAW'
  b_newdn    = 'X'
  b_cleanup  = 'X'
  b_distribute = 'X'
*  b_distribute_sync = 'X'
*  b_pinchg  = 'X'

EXCEPTIONS

  file_load_failed = 1
  storing_failed   = 2
  authority_missing = 3
  OTHERS          = 4.

* -----

IF sy-subrc EQ 0.

CALL FUNCTION 'SSFPSE_UPDATED'

EXPORTING

  psename      = lv_psename
*  b_host      = 'X'          " Yes/No
*  b_instanceid = 'X'        " Yes/No

EXCEPTIONS

  authority_missing = 1          " No authorization for this function
  ssf_krn_nomemory = 2          " Main Memory Not Sufficient

```

```

ssf_krn_nossflib      = 3

ssf_krn_invalid_par  = 4      " An SSF Parameter is Erroneous

ssf_krn_invalid_parlen = 5      " Length of an SSF Parameter is Erroneous

ssf_krn_error        = 6

database_failed      = 7

unknown_error        = 8      " Unknown SSF error

OTHERS                = 9.

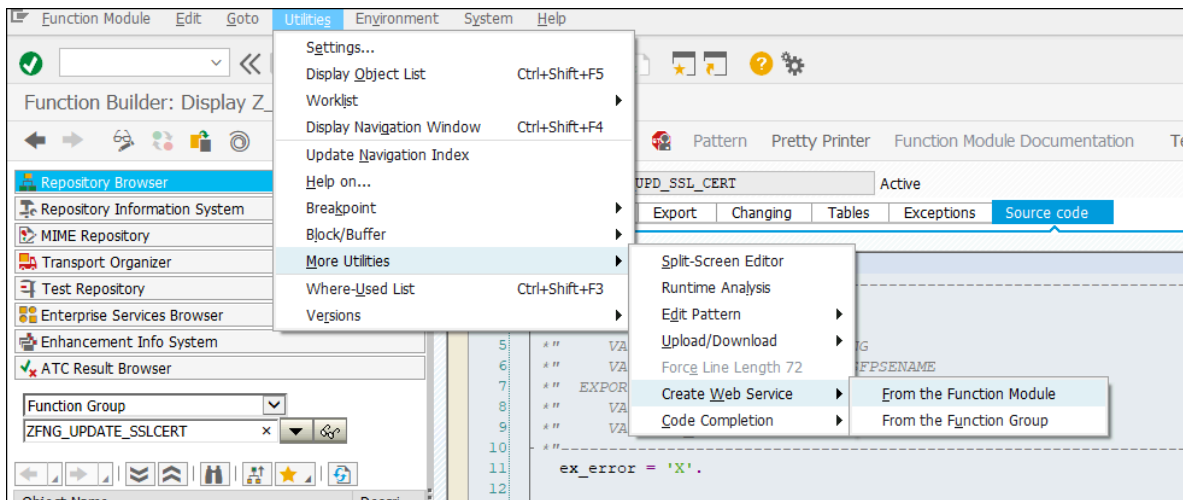
ENDIF.

* -----
* Delete temporary PSE file
* -----

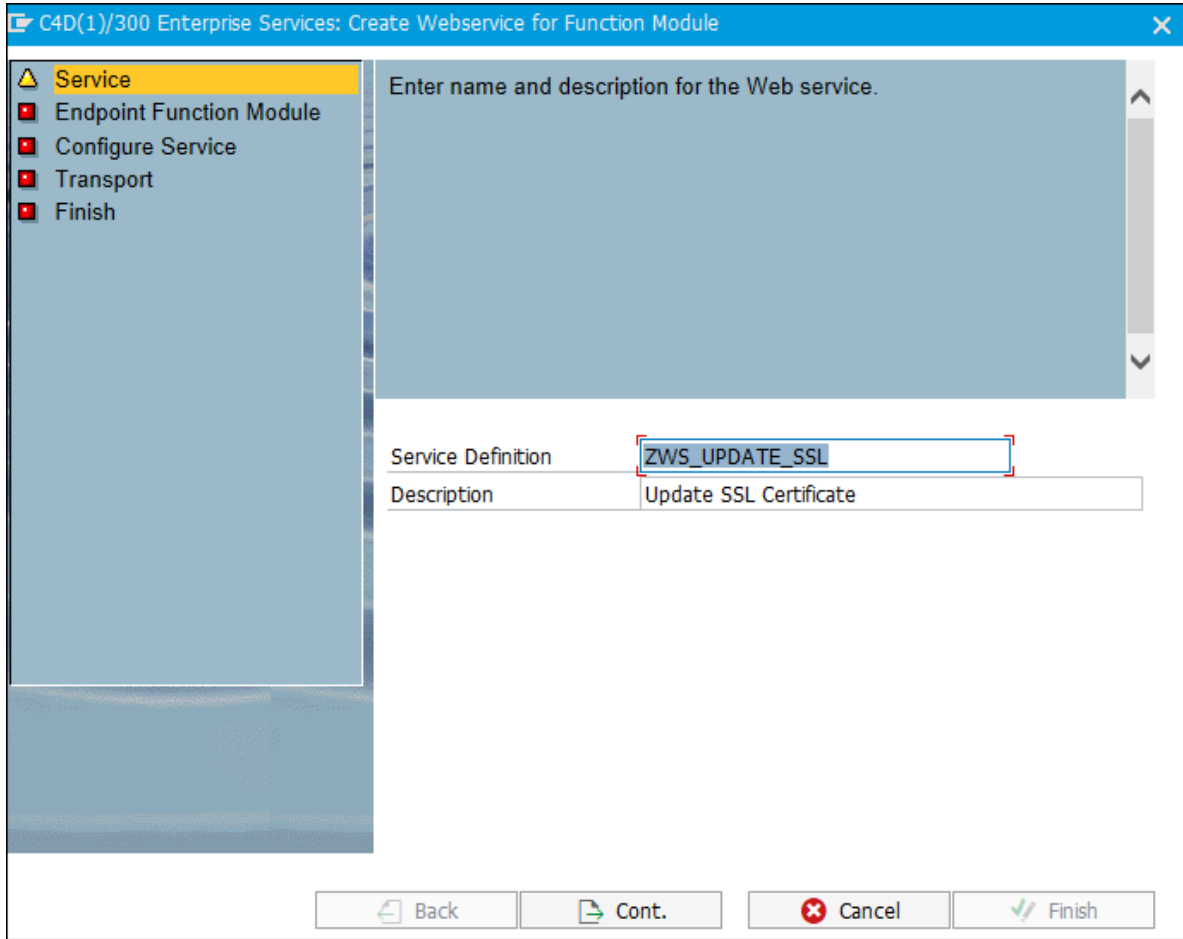
DELETE DATASET lv_fname. "temporary PSE file (SSFPSE_LOAD)

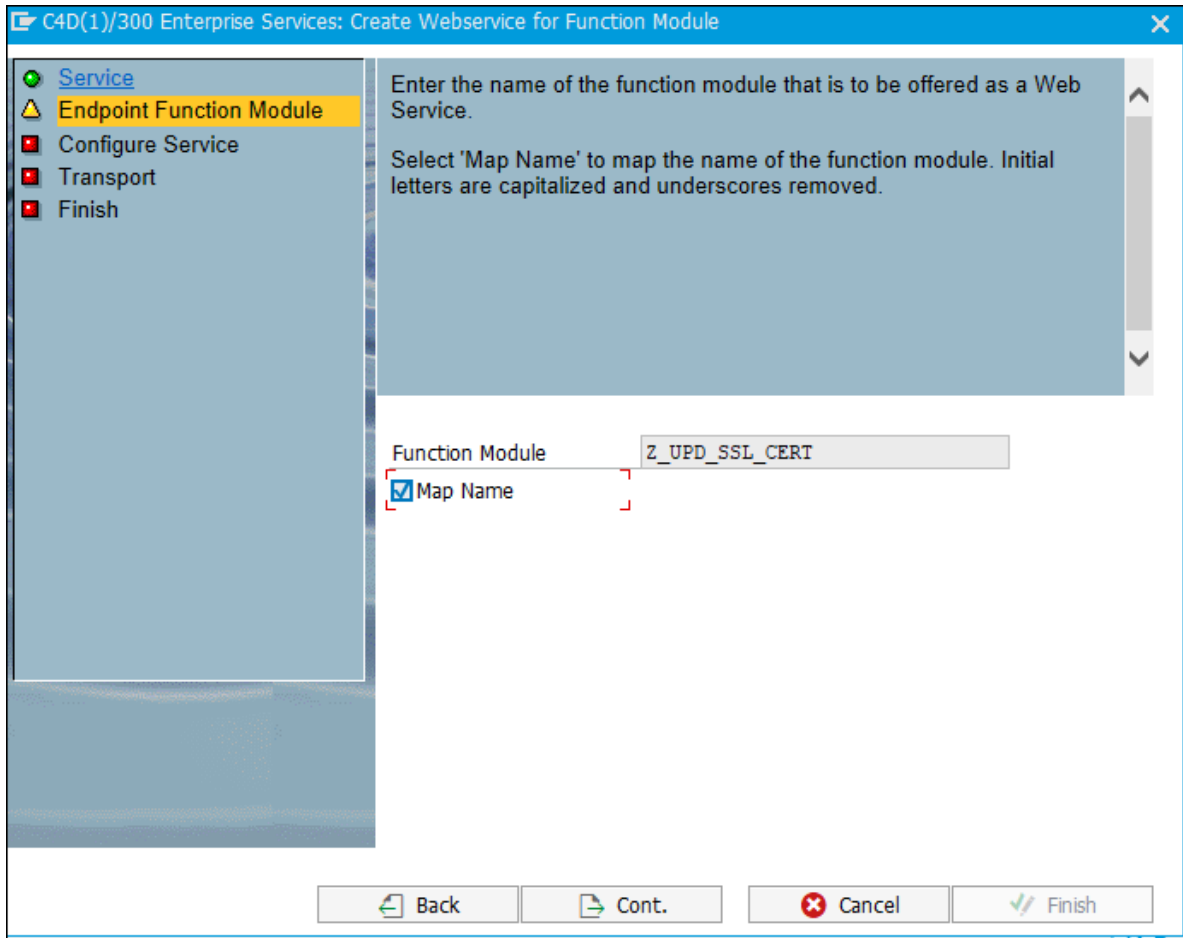
ENDFUNCTION.
    
```

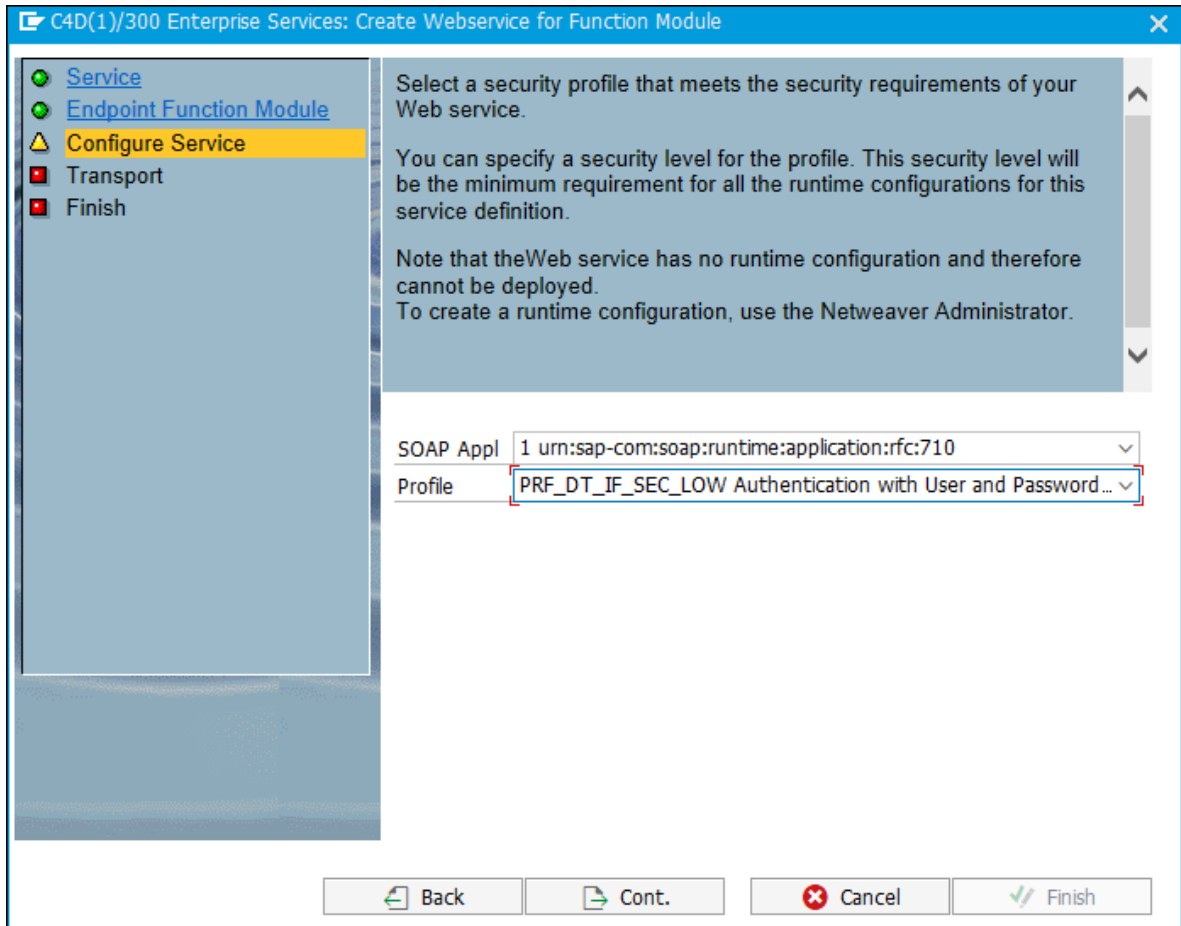
8. Create **Service Definition** as shown below
  - a. Open the function module **Z\_UPD\_SSL\_CERT**.
  - b. Create service definition from below path.



- c. Follow the screens below to create the service definition.







- d. Specify the **Package and Transport Request** and click **Finish**.
  - e. Activate the service definition.
9. Create Binding in SOA as follows:
- a. Run the Transaction **SOAMANAGER**.
  - b. Click **Web Service Configuration**.

The screenshot shows the SAP SOA Management (C4D;300) interface. At the top, there is a dark blue header with the SAP logo and the text "SOA Management (C4D;300)". Below the header, there is a navigation bar with three tabs: "Service Administration" (which is underlined), "Technical Administration", and "Logs and Trace". The main content area lists several menu items:

- Identifiable Business Context**  
Display and maintain Identifiable Business Contexts (IBCs)
- Identifiable Business Context Reference**  
Display and maintain Identifiable Business Contexts references (IBC reference)
- Design Time Cache**  
Display central design time cache
- Web Service Configuration** (highlighted in yellow)  
Configure service definitions, consumer proxies and service groups
- Simplified Web Service Configuration**  
Configure service definitions for Web service consumers with limited capabilities

- c. Enter the service definition name created and click **Search**.
- d. Click on the service definition.

The screenshot shows the SAP Web Service Configuration interface. At the top, it says "Web Service Configuration (C4D;300)". Below that, there are two tabs: "Design Time Object Search" (selected) and "Configuration Search". Under "Design Time Object Search", there is a "Search criteria" section with the following fields: "Object Type" (dropdown), "is" (dropdown), "All" (dropdown), "Object Name" (dropdown), "is" (dropdown), and "ZWS\_UPDATE\_SSL" (text input). There is also a "Maximum Number of Results" field set to "100". Below these are buttons for "Search", "Clear values", and "Reset search criteria". A "Save" button is partially visible on the right. Below the search criteria is a "Search Result" section with a table:

Internal Name	Type	Name
ZWS_UPDATE_SSL	Service Definition	ZWS_UPDATE_SSL

e. Click **Create Service**.

The screenshot shows the "Details of Service Definition: ZWS\_UPDATE\_SSL" page. It has tabs for "Overview", "Configurations" (selected), "Classifications", and "Details". Under "Define Services and Bindings", there are several buttons: "Create Service" (highlighted in yellow), a plus icon, a minus icon, "Activate", "Deactivate", "Delete", "Republish", and "Display as List". Below the buttons is a table with the following columns: "Service/Binding", "Actions", "State", and "Description".

Service/Binding	Actions	State	Description
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			
<input type="checkbox"/>			

f. Enter the details as shown below and click **Next** on each step. On the final step, click **Finish**.

**SAP** Web Service Configuration (C4D;300)

### Configuration of New Binding for Service Definition 'ZWS\_UPDATE\_SSL'

1 — 2 — 3 — 3

**Service and Binding Name**    Provider Security    SOAP Protocol    Operation Settings

Back   **Next**   Finish   Cancel

#### Service Information

\* Service Name:

Service Description Text:

#### Binding Information

\* New Binding Name:

**Message Level Security**

None

Symmetric Message Signature and Encryption

Asymmetric Message Signature

Asymmetric Message Signature and Encryption

Secure Conversation

Extended Signature and Header Protection

**Authentication Settings**

Authentication Level: Basic

**Authentication Method**

No Authentication

**Transport Channel Authentication**

User ID/Password

X.509 SSL Client Certificate

Single Sign On using SAP Assertion Ticket

**SAP** Web Service Configuration (C4D;300)

Configuration of New Binding for Service Definition 'ZWS\_UPDATE\_SSL'

1 — 2 — 3 — 3

Service and Binding Name    Provider Security    **SOAP Protocol**    Operation Settings

Back   **Next**   Finish   Cancel

**Transport Binding**

Alternative Access URL:

Calculated Access URL:

Calculated Protocol: HTTP

Make Local Call: No Call in Local System

State Management Timeout:

**Message Attachment Handling**

Process Attachments: No

**SAP** Web Service Configuration (C4D;300)

Configuration of New Binding for Service Definition 'ZWS\_UPDATE\_SSL'

1 — 2 — 3 — 3

Service and Binding Name    Provider Security    SOAP Protocol    **Operation Settings**

Back   Next   **Finish**   Cancel

Operation
<input checked="" type="radio"/> ZUpdSslCert
<input type="radio"/>
<input type="radio"/>
<input type="radio"/>

g. Once the Binding is generated, follow the steps shown below to get the URL.

**SAP** Web Service Configuration (C4D;300)

### Details of Service Definition: ZWS\_UPDATE\_SSL

Overview **Configurations** Classifications Details

#### Define Services and Bindings

<input type="checkbox"/>	Service/Binding	Actions	State
<input type="checkbox"/>	▼ ZWS_UPDATE_SSL		Activ
<input checked="" type="checkbox"/>	ZBIND_UPDATE_SSL		
<input type="checkbox"/>			
<input type="checkbox"/>			

#### WSDL Generation for Binding: ZBIND\_UPDATE\_SSL

SAP Assertions: 
 WSP Version: 
 SOAP Action:

Security Assertions: 
 WSP Style: 
 Show Extensibility:

WSDL Section: 
 SOAP Version:

WSDL Version: 1.1
 SOAP Style:

#### Options for WSDL Access and URLs

Standard  
 Alternative URL

Alternate Host:  Alt. Port (http):   
 Meta Data Protocol:  Alt. Port (https):

#### WSDL Generation

**WSDL URL for Binding:** [http://\[redacted\]8001/sap/bc/srt/wsd/ftv\\_10002A101AD1/bndg\\_url/sap/bc/srt/rfc/sap/zws\\_update\\_ssl/300/zws\\_update\\_ssl/zbind\\_update\\_ssl?sap-client=300](http://[redacted]8001/sap/bc/srt/wsd/ftv_10002A101AD1/bndg_url/sap/bc/srt/rfc/sap/zws_update_ssl/300/zws_update_ssl/zbind_update_ssl?sap-client=300)

## Chapter 2: More Information

For the latest, most complete information about known and fixed issues with the AppViewX modules, see the latest revision of the release notes.

To access Software Release Notifications for AppViewX Releases, visit our Help center at <https://help.appviewx.com/home>. You need to log in to your AppViewX account. From the Help center, search by the specific release number or navigate to Release Portal and choose the release, for example, v20.3.0.

### Documentation Feedback

We request you to provide feedback, comments, and suggestions so that we can improve the documentation. You can send your comments to [tech-documentation@appviewx.com](mailto:tech-documentation@appviewx.com)

If you are preferred to send feedback through e-mail, be sure to include the following information with your comments:

- Document or topic name
- URL or page number
- Software release version (if applicable).

### Requesting Technical Support

Technical product support is available through AppViewX help support center, request to send an email to [help@appviewx.com](mailto:help@appviewx.com)

### Self-Help Online Tools and Resources

For quick and easy problem resolution, AppViewX is designed an online self-service portal called the help support center that provides you with the following features:

- Find help support center: <https://help.appviewx.com/home>
- Find product technical documentation: <https://help.appviewx.com/documentation>
- Find solutions and answer questions using our Knowledge Base: <https://internalkb.appviewx.com/knowledge-base>
- Download the latest versions of software: <https://release.appviewx.com>