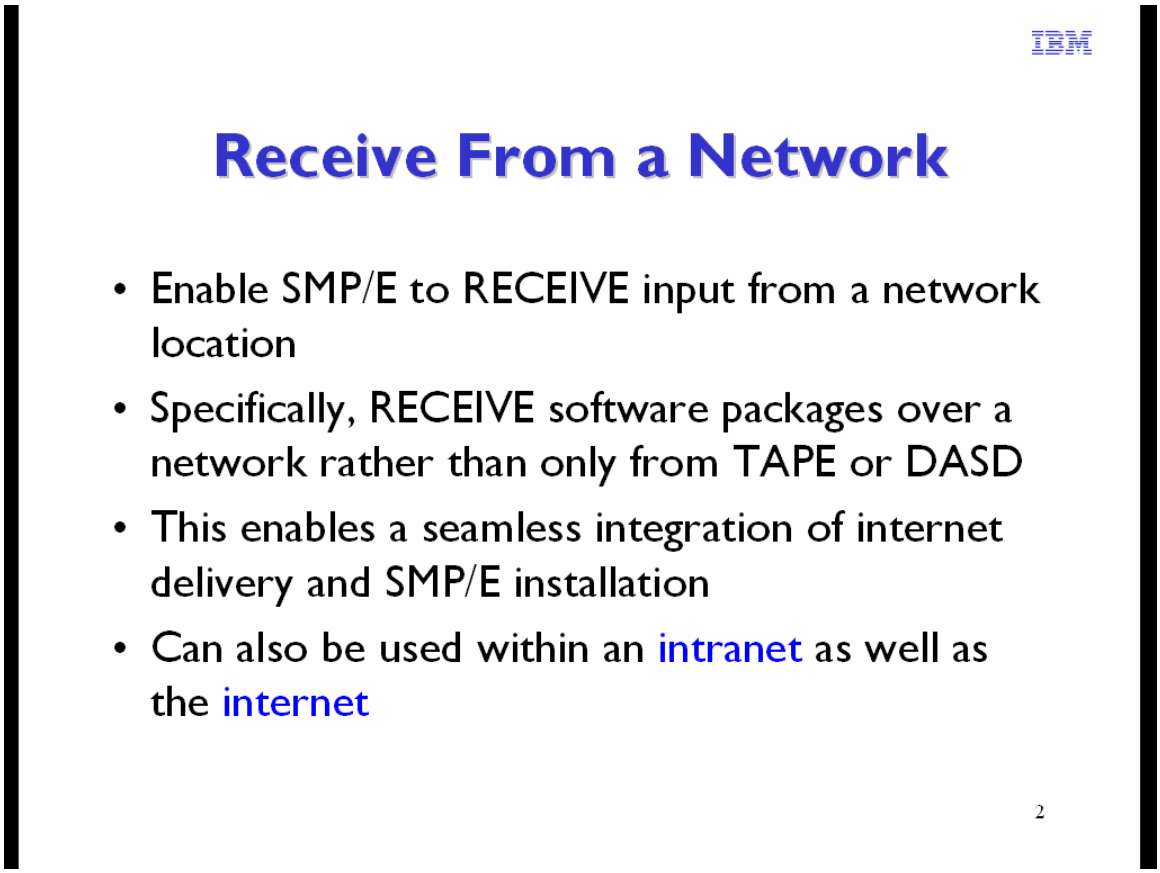
Internet delivery of software products and service is gaining popularity for several reasons. SMP/E provides the capability to receive products and service directly from the internet to the host. This session will step you through the SMP/E RECEIVE FROMNETWORK function beginning with setting up its prerequisites through SMP/E receive processing.



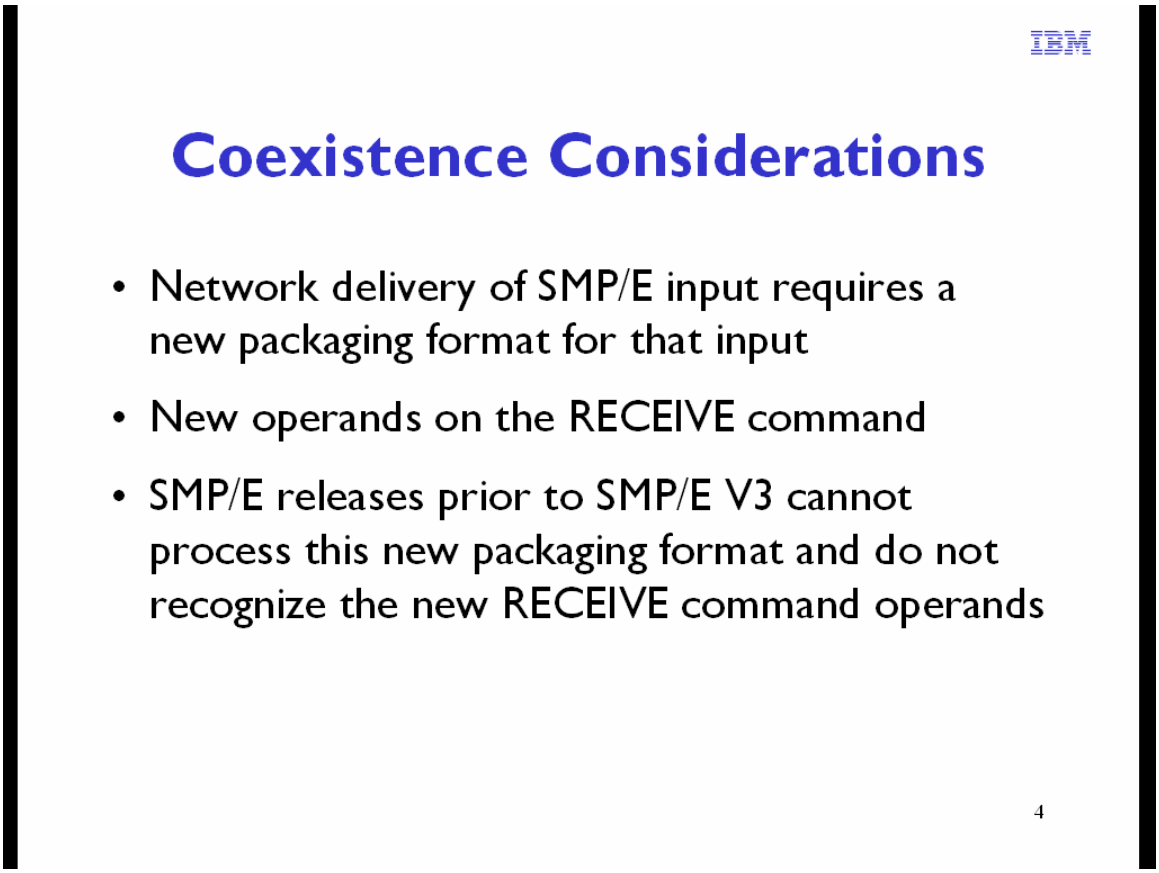
Network Delivery of SMP/E Input
SMP/E V3R1 and higher can receive input from a network server, in addition to tape and DASD. This enables the delivery of SMP/E-installable products and service over the internet or an intranet. By installing software directly from a network source, SMP/E enables a more seamless integration of electronic software delivery and installation. This reduces the tasks and time required to install software delivered electronically.

IBM

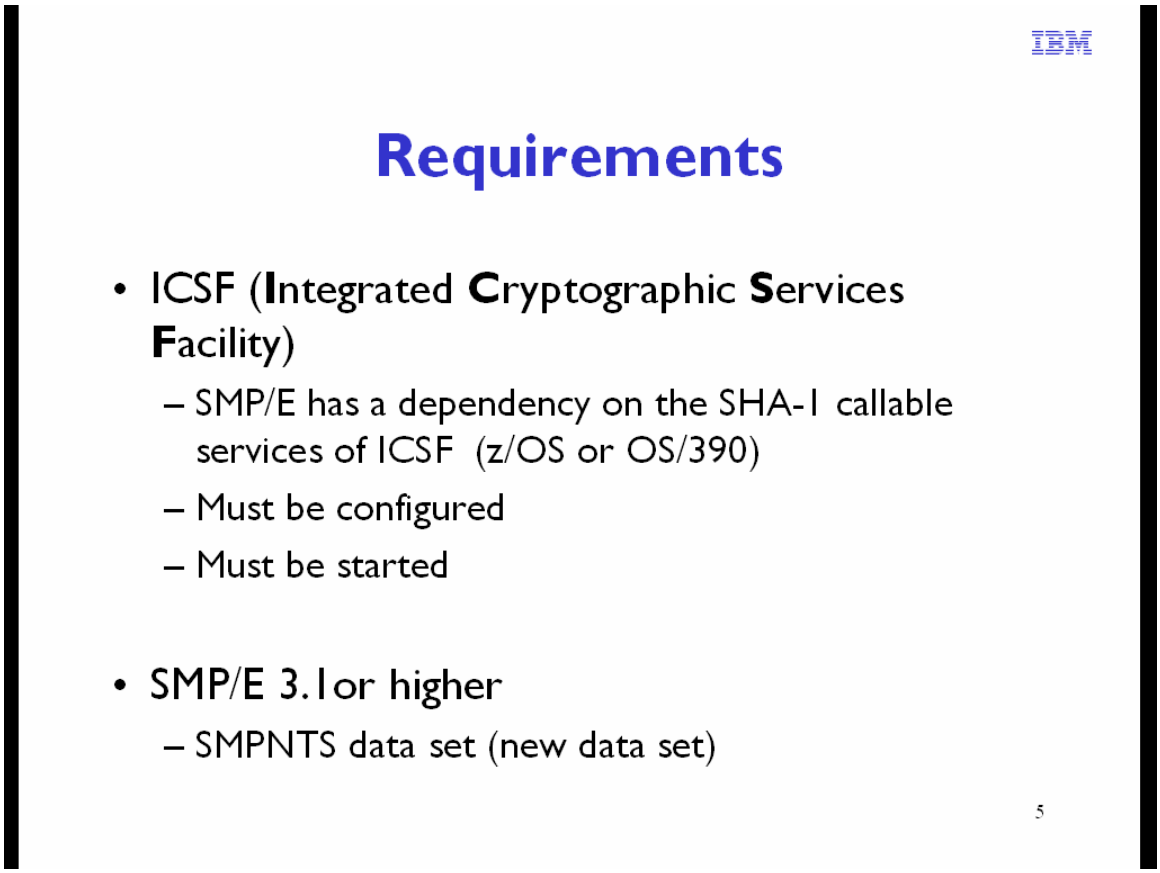
Presentation Focus

- There are several ways to receive products and service from the internet
  - Download from an IBM server directly to your host using JCL (or batch FTP)
  - Store on a workstation and then forward to the host
    - Using Download Director
    - FTP through your browser
- The focus of this presentation is primarily on the SMP/E command RECEIVE FROMNETWORK

3

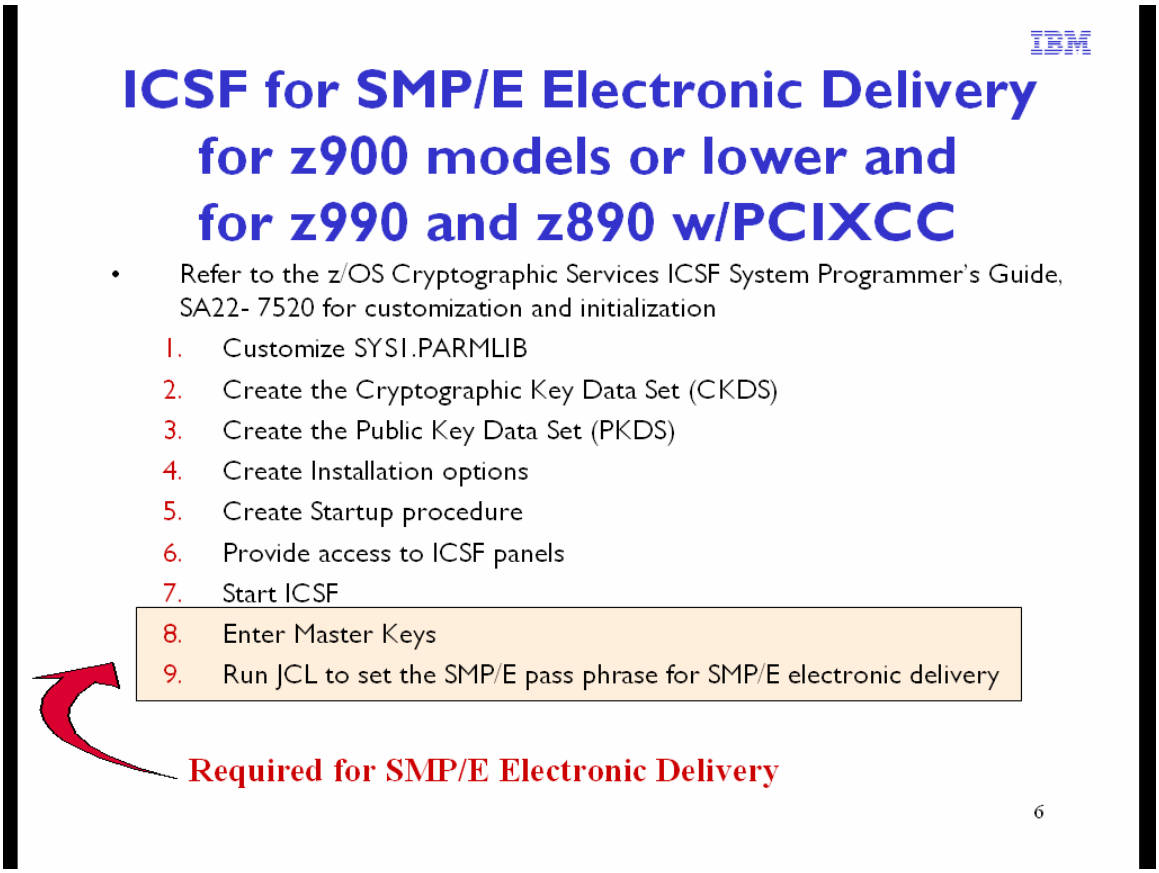


Network delivery of SMP/E input requires a new packaging format for that input and new operands on the RECEIVE command. SMP/E releases prior to z/OS SMP/E cannot process this new packaging format and do not recognize the new RECEIVE command operands.



Refer to the *z/OS Cryptographic Services ICSF System Programmer's Guide*, SA22-7520 for customization and initialization of ICSF.

The minimum SMP/E release is 3.1. SMP/E is a separately orderable product as well as an optional base element in z/OS. There are several new SMP/E data sets with this level of SMP/E. Specifically the SMPNTS.



Refer to the *z/OS Cryptographic Services Integrated Cryptographic Service Facility Programmer's Guide* for complete details.

Customize PARMLIB

The installation options data set is generally stored in SYS1.PARMLIB. If your administrator does not have access to SYS1.PARMLIB, you need to use another data set instead. Update the data set you are using as follows:

Add CEE.SCEERUN and CSF.SCSFMOD0 to the LNKLST concatenation This adds the ICSF library to the z/OS library search. The following is an example of an ICSF entry to the LNKLST concatenation. CSF.SCSFMOD0

APF authorize CSF.SCSFMOD0, if LNKAUTH=APFTAB. The following is an example of an ICSF entry for APF authorization. APF ADD DSNAME(CSF.SCSFMOD0) VOLUME(******)

In the IKJTSOxx parameter, add CSFDAUTH and CSFDPKDS as a value in the AUTHPGM parameter list and in the AUTHTSF parameter list. The following is an example of an ICSF entry in the IKJTSOxx member.

Create the CKDS (Cryptographic Key Data Set)

The CKDS must be a key-sequenced data set with a fixed record length of 252 bytes. Allocate the CKDS on a permanently resident volume.

1. Determine the amount of primary space you need to allocate for the CKDS. This should reflect the total number of entries you expect the data set to contain originally.
2. Determine the amount of secondary space to allocate for CKDS. This should reflect the total number of entries you expect to add to the data set. For detailed information about calculating space for a VSAM data set, see z/OS DFSMS Access Method Services for Catalogs.

3. Create an empty VSAM data set to use as the CKDS. ICSF provides a sample job to define the CKDS in member CSFCKDS of SYS1.SAMPLIB

Create the PKDS (Public Key Data Set)
The PKDS must be allocated and specified on the PKDSN parameter of the options data set when you first start ICSF. ICSF support for the PCI Cryptographic Coprocessor or PCI X Cryptographic Coprocessor requires a PKDS. Since ICSF can not tell if a PCI Cryptographic Coprocessor will be added, it requires the PKDS to be available at start up.

The PKDS must be a key-sequenced data set with variable length records. Allocate the PKDS on a permanently resident volume.

1. Determine the amount of primary space you need to allocate for the PKDS. This should reflect the total number of entries you expect the data set to contain originally. The PKDS will contain both public and private PKA keys.
2. Determine the amount of secondary space to allocate for the PKDS. This should reflect the total number of entries you expect to add to the data set. For detailed information about calculating space for a VSAM data set, see z/OS DFSMS Access Method Services for Catalogs.
3. Create an empty VSAM data set to use as the PKDS. Use the AMS DEFINE CLUSTER command to define the data set and to allocate its space. ICSF provides a sample job to define the PKDS in member CSFPKDS of SYS1.SAMPLIB.
4. Allocate a disk copy of the PKDS by defining a VSAM cluster. ICSF provides a sample job CSFPKDS of SYS1.SAMPLIB.

Create the Installation Options Data Set
Steps to create the Installation Options Data Set
The *installation options data set* is a file that you create that contains installation options. It becomes active when you start ICSF.

- The installation options data set can be a member of PARMLIB, a partitioned data set, a member of a partitioned data set, or a sequential data set.
- The format of each record in the data set must be fixed length or fixed block length.
- A physical line in the data set is 80 characters long. The system ignores any characters in positions 72 to 80 of the line.
- A logical line is one or more physical lines. You can group physical lines into a logical line by placing a comma at the end of the information. Only a comment can appear after the comma. The system ignores any other information between the comma and column 71.
- Continuation causes the next physical line to append immediately following the comma. The system removes all leading blanks on the next physical line.
- You can delimit comments by /* and */ and include them anywhere within the text. A comment cannot span physical records. The system removes comments from a logical line before parsing it. It ignores physical lines that contain only comments.
- Specify only one option setting or keyword on a logical line. (If you specify more than one, the system ignores all but the last one on the line. The system reports syntax errors, but the errors do not cause it to stop interpreting the file.)

ICSF provides two sample installation options data sets. These sample data sets use the recommended values for each option. They are members CSFPRM00 and CSFPRM01 in SYS1.SAMPLIB.

Create the ICSF Startup Procedure

ICSF provides two job control language programs. You can use this code as the basis for your startup procedure.

- member CSF in SYS1.SAMPLIB
- member CSFSTART in SYS1.SAMPLIB (batch setup for SMP/E)
  Store this startup PROC in SYS1.PROCLIB (or another suitable library). For SMP/E this proc can be copied to the CPAC.PROC data set, and the data set name (SYS1.PARMLIB) can be copied to match the name you chose for the CPAC.PARMLIB data set (in which CSFPRM01 would be placed). This would work for Server-Pac. Outside of Server-Pac, you need to copy and edit CSFSTART.

1. Change or use the sample startup procedure according to your needs.
2. Store your startup procedure in SYS1.PROCLIB (or another suitable library) with a member name of your choice. (Depending on installation standards, possible names include CSF, CSFPROD, CSFTEST, and CRYPTO.)
3. If you use Security Server (RACF), you may need to update the RACF Started Procedure Table if you define a new started task:

Provide access to the ICSF panels (OPTIONAL)

To provide a way for the administrator to access the ICSF panels, you can create an ICSF option on the ISPF Primary Option Menu. Access the code for the ISPF Primary Option Menu panel body and perform the following steps:

1. Under the % OPTION ===> _ZCMD line, add the following line: % <option value> - ICSF Panels You can specify either a letter or number for the option value. Do not use an option value that already exists in the menu.
2. On the &ZSEL= TRANS(&ZQ line, add the following information: <option value>,'PANEL(CSF@PRIM)' The option value should be the same value as the option value you chose to use in the preceding step.

When you access the ISPF Primary Option Menu panel, the ICSF panels option appears on the menu. You can choose the ICSF option value to access the ICSF panels.

You must also update the logon procedure that is used by ICSF administrators who will use the ICSF panels.

An alternate method to access the ICSF panels is to use ISPF LIBDEF.

Steps to start ICSF for the first time

Before using ICSF you must initialize it, which you are now ready to do.

1. Enter the START command and the startup procedure name. When you start ICSF, you specify the name of the ICSF startup procedure you created.
2. When you start ICSF for the first time, you will see different messages depending on your system hardware.

Master Key Initialization for SMP/E Only – CCF Systems Only

Run the JCL to set the SMP/E pass phrase for SMP/E electronic delivery only.

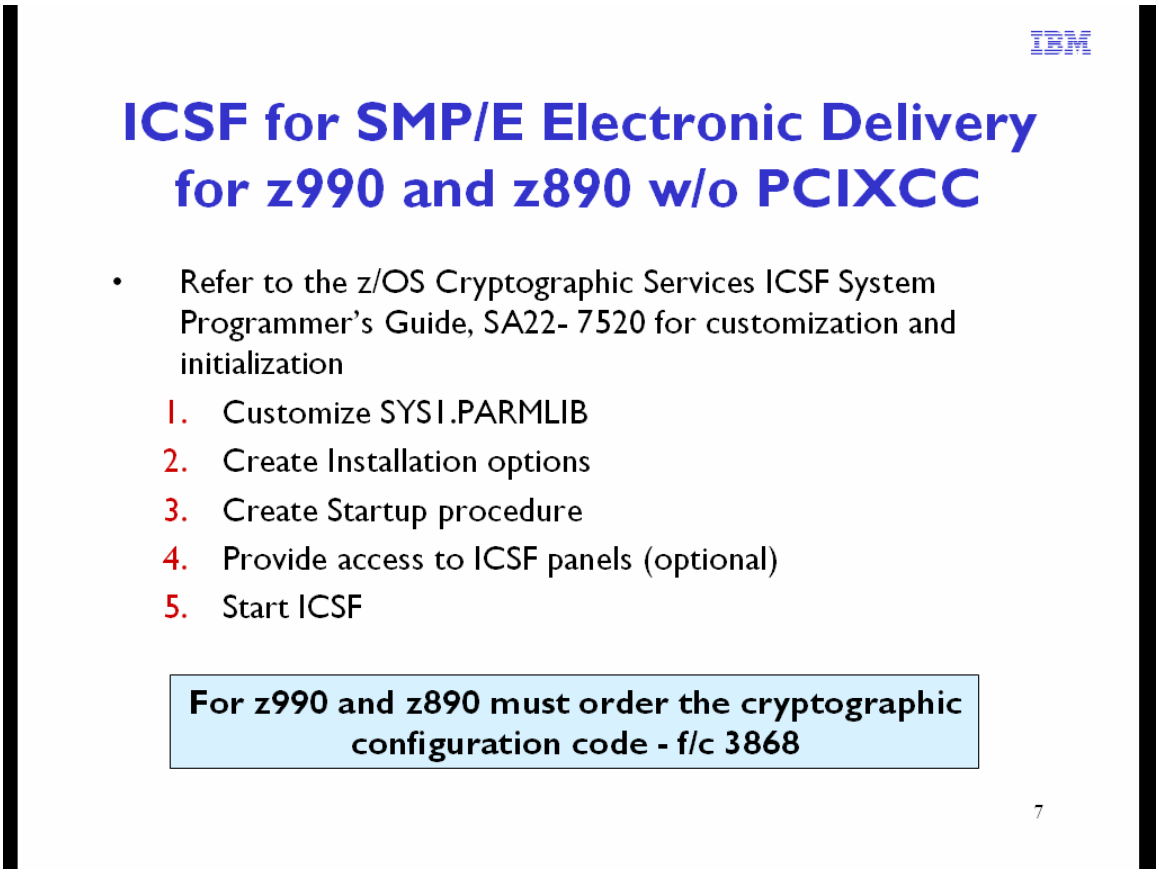
The JCL uses a pass phrase value to load the DES and PKA master keys. The DES and PKA master keys will be set in the Cryptographic Coprocessor Feature. *Change This Pass Phrase* is the default pass phrase. The entry point is CSFEUTIL and will have 2 or (optionally) 3 parameters. The first parameter must be the CKDS name. The second parameter (optional) is the pass

phrase. The last parameter is the function PPINIT. If you do not use the default pass phrase and create your own:

- It must be sixteen to sixty-four bytes in length.
- Any EBCDIC character is allowed.
- Leading and trailing blanks will be removed.
- Embedded blanks are allowed

In order to successfully run the CSFSETMK job, determine if the following services are RACF protected in the CSFSERV class. If the services below are not RACF protected in the CSFSERV class, then nothing needs to be done. If the services are protected in the CSFSERV class, then the issuer of the CSFSETMK JCL must be permitted to the profile for each service.

- CSFOWH
- CSFPMCI
- CSFCMK
- CSFREFR



Customize PARMLIB

The installation options data set you will create is generally stored in SYS1.PARMLIB. If your administrator does not have access to SYS1.PARMLIB, you need to use another data set instead. Update the data set you are using as follows:

1. Add CEE.SCEERUN and CSF.SCSFMOD0 to the LNKLST concatenation This adds the ICSF library to the z/OS library search. The following is an example of an ICSF entry to the LNKLST concatenation.
   CSF.SCSFMOD0
2. APF authorize CSF.SCSFMOD0, if LNKAUTH=APFTAB. The following is an example of an ICSF entry for APF authorization.
   APF ADD DSNAME(CSF.SCSFMOD0) VOLUME(******)
3. In the IKJTSOxx parameter, add CSFDAUTH and CSFDPKDS as a value in the AUTHPGM parameter list and in the AUTHTSF parameter list.

Steps to create the Installation Options Data Set

The *installation options data set* is a file that you create that contains installation options. It becomes active when you start ICSF.

- The installation options data set can be a member of PARMLIB, a partitioned data set, a member of a partitioned data set, or a sequential data set.
- The format of each record in the data set must be fixed length or fixed block length.
- A physical line in the data set is 80 characters long. The system ignores any characters in positions 72 to 80 of the line.
- A logical line is one or more physical lines. You can group physical lines into a logical line by placing a comma at the end of the information. Only a comment can appear after

the comma. The system ignores any other information between the comma and column 71.
- Continuation causes the next physical line to append immediately following the comma. The system removes all leading blanks on the next physical line.
- You can delimit comments by /* and */ and include them anywhere within the text. A comment cannot span physical records. The system removes comments from a logical line before parsing it. It ignores physical lines that contain only comments.
- Specify only one option setting or keyword on a logical line. (If you specify more than one, the system ignores all but the last one on the line. The system reports syntax errors, but the errors do not cause it to stop interpreting the file.)

ICSF provides two sample installation options data sets. These sample data sets use the recommended values for each option. They are members CSFPRM00 and CSFPRM01 in SYS1.SAMPLIB.

Create the ICSF Startup Procedure

ICSF provides two job control language programs. You can use this code as the basis for your startup procedure.

- member CSF in SYS1.SAMPLIB
- member CSFSTART in SYS1.SAMPLIB (batch setup for SMP/E)
  Store this startup PROC in SYS1.PROCLIB (or another suitable library). For SMP/E this proc can be copied to the CPAC.PROC data set, and the data set name (SYS1.PARMLIB) can be copied to match the name you chose for the CPAC.PARMLIB data set (in which CSFPRM01 would be placed). This would work for Server-Pac. Outside of Server-Pac, you need to copy and edit CSFSTART.

4. Change or use the sample startup procedure according to your needs.
5. Store your startup procedure in SYS1.PROCLIB (or another suitable library) with a member name of your choice. (Depending on installation standards, possible names include CSF, CSFPROD, CSFTEST, and CRYPTO.)
6. If you use Security Server (RACF), you may need to update the RACF Started Procedure Table if you define a new started task:

Provide access to the ICSF panels (OPTIONAL)

To provide a way for the administrator to access the ICSF panels, you can create an ICSF option on the ISPF Primary Option Menu. Access the code for the ISPF Primary Option Menu panel body and perform the following steps:

3. Under the % OPTION ===> _ZCMD line, add the following line: % <option value> - ICSF Panels You can specify either a letter or number for the option value. Do not use an option value that already exists in the menu.
4. On the &ZSEL= TRANS(&ZQ line, add the following information: <option value>,'PANEL(CSF@PRIM)' The option value should be the same value as the option value you chose to use in the preceding step.

When you access the ISPF Primary Option Menu panel, the ICSF panels option appears on the menu. You can choose the ICSF option value to access the ICSF panels.

You must also update the logon procedure that is used by ICSF administrators who will use the ICSF panels.

An alternate method to access the ICSF panels is to use ISPF LIBDEF.

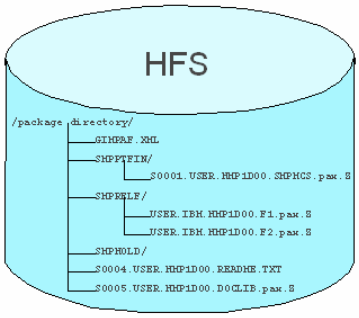
Steps to start ICSF for the first time

Before using ICSF you must initialize it, which you are now ready to do.

3. Enter the START command and the startup procedure name. When you start ICSF, you specify the name of the ICSF startup procedure you created.
4. When you start ICSF for the first time, you will see different messages depending on your system hardware.

SMPNTS Data Set

The SMPNTS is a directory structure and associated files contained in the hierarchical file system used for temporary storage of network transported packages received during SMP/E RECEIVE processing



8

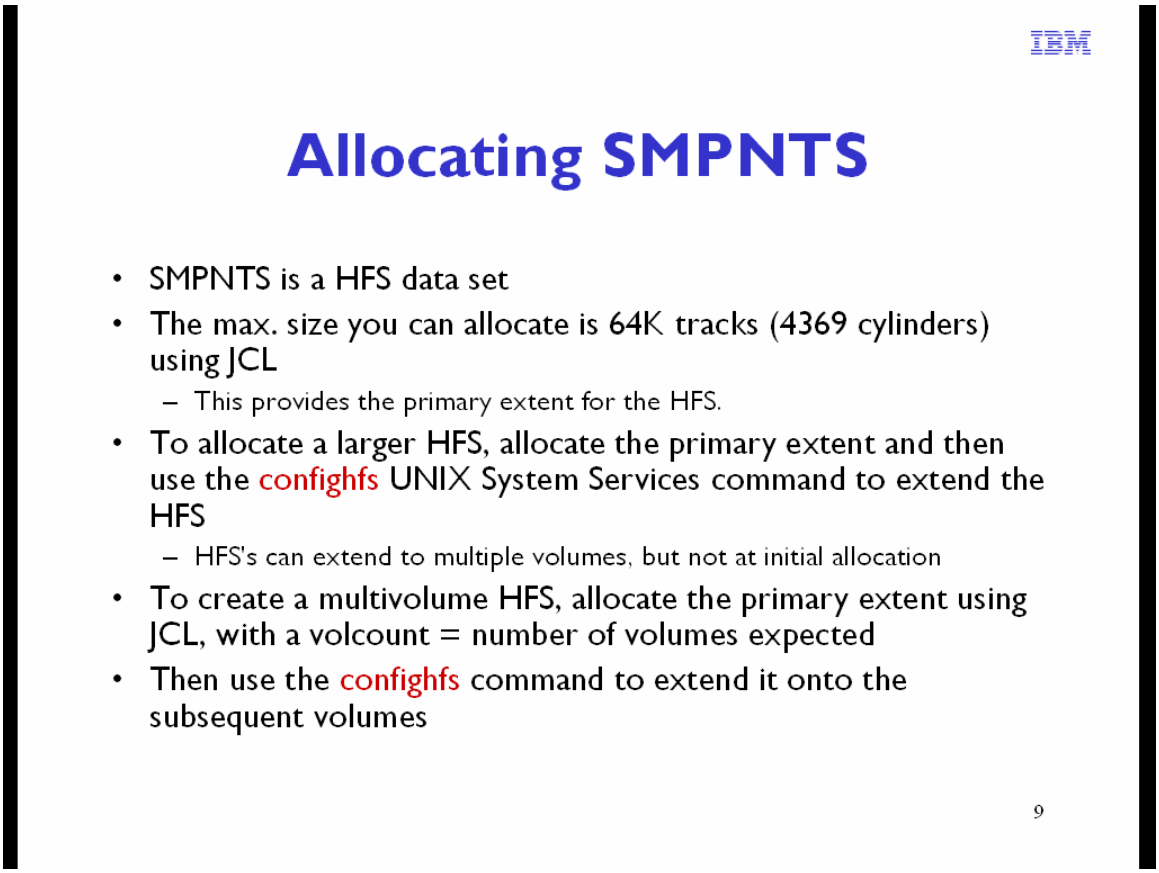
The SMPNTS is an existing directory within the hierarchical file system. The hierarchical file system directory name is concatenated with the appropriate subdirectories and file names to create complete pathnames. The directory name can be from 1 to 255 characters. The directory name must begin and end with a slash (/). In addition to the required delimiters (/), a directory name must also be enclosed in single apostrophes (') if any of the following is true:

- The directory name contains lowercase alphabetic characters
- The directory name contains a character that is not uppercase alphabetic, numeric, or national ($, #, or @), slash (/), plus (+), hyphen, period, or ampersand (&).

The apostrophes must be outside the required delimiters, as in '/directory name/', not /'directory name'/. The single apostrophes used to enclose the directory name (the delimiters) do not count as part of the 255 character limit. Any apostrophes specified as part of the directory (not the delimiters) must be doubled. Double apostrophes count as two characters in the 255-character limit. The directory name can include characters through X'40' and X'FE'. **Do not use symbolic substitution.**

Device: Direct access only.

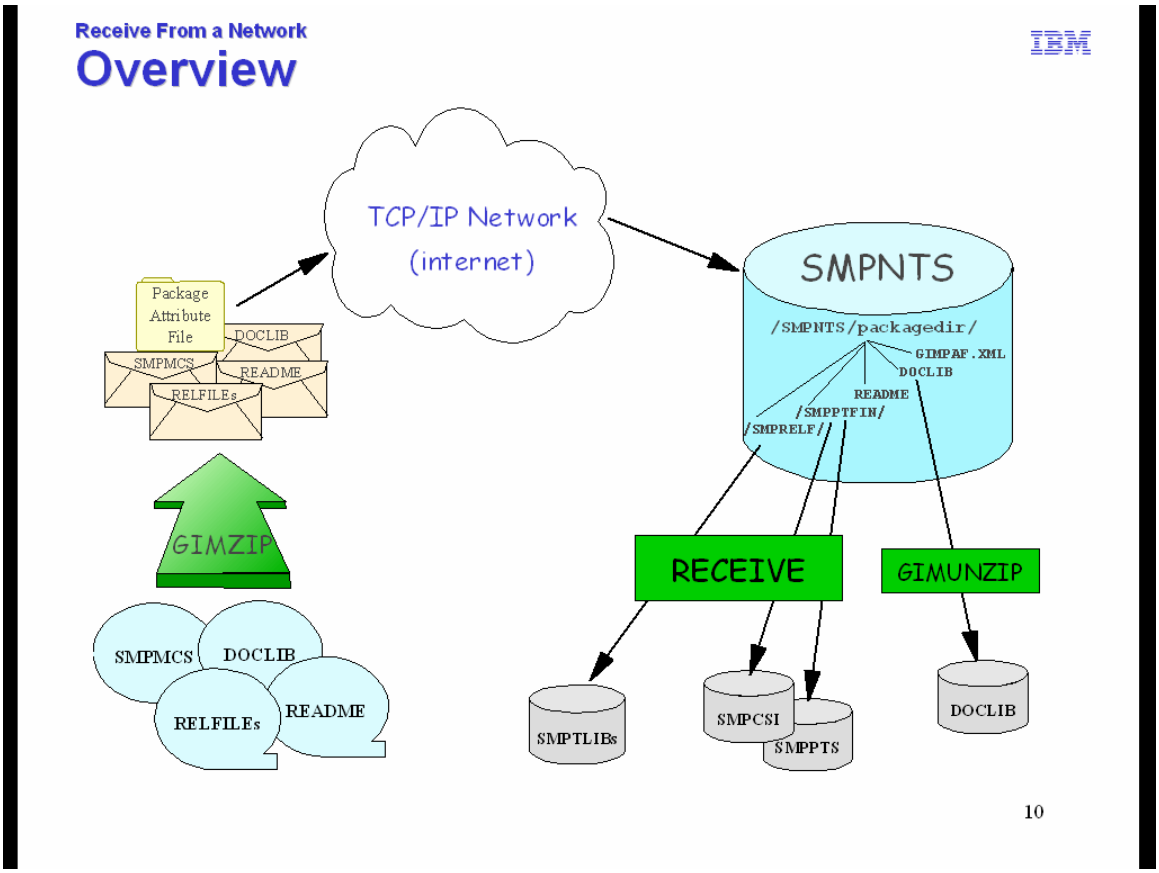
Notes:
The SMPNTS can be defined to SMP/E only with a DD statement or a DDDEF. Do not allocate the SMPNTS as anything other than a directory in the hierarchical file system. The size of the SMPNTS directory depends on the size of the packages received from the network and stored there.



Allocating the HFS

```
//*This job allocates a large HFS file. Note the following restrictions:
//*
//* 1. The max. size you can allocate is 64K tracks (4369 cylinders)
//*    using JCL. This provides the primary extent for the HFS.
//*    To allocate a larger HFS, allocate the primary extent and
//*    then use the confighfs Unix Services command to extend the
//*    HFS.
//* 2. HFS's can extend to multiple volumes, but not at initial
//*    allocation. To create a multivolume HFS, allocate the primary
//*    extent using JCL, with a volcount = number of volumes expected.
//*    Then use the confighfs command to extend it onto the subsequent
//*    volumes. Note the following:
//*    volumes. Note the following:
//*    1. Multivolume HFS's are ONLY supported on SMS managed DASD.
//*    2. The confighfs command does not have the 64K track limitation,
//*       so the whole subsequent volumes can be allocated in one
//*       extent.
//* 3. The confighfs command is a Unix Services command. You must
//*    run it from the shell, or using BPXBATCH. Also, it is found
//*    in /usr/lpp/dfsms/bin, which may not be in your standard
//*    PATH. If so, either update your profile to add it to the
//*    path, or invoke it explicitly (/usr/lpp/dfsms/bin/confighfs).
//*
//*--------------------------------------------------------------------
//ALLOC EXEC PGM=IEFBR14
//*--------------------------------------------------------------------
//* Allocate the primary extent via JCL, with a volcount.
//*--------------------------------------------------------------------
//HFS   DD  DISP=(,CATLG),UNIT=SYSDA,DSN=<hfs name>,
```

```
//       SPACE=(CYL,(10,10,1)),DSNTYPE=HFS,
//       VOL=(,,,3)
//TSO    EXEC TSOBATCH
//*-------------------------------------------------------------------
//*-------------------------------------------------------------------
//* Run TSO batch to mount the HFS above and add space to it using
//* the confighfs command.
//*
//* In this example, we are adding a 10 cylinder extent to the HFS
//* To extend to another volume, use the -xn option.
//*
//*-------------------------------------------------------------------
//SYSIN  DD *
MOUNT FILES('<hfs name>') TYPE(HFS) MOUNTP('<valid_mountpoint>')
OSHELL /usr/lpp/dfsms/bin/confighfs -x[n] __c <valid_mountpoint>
UNMOUNT FILES('<hfs name>')
```



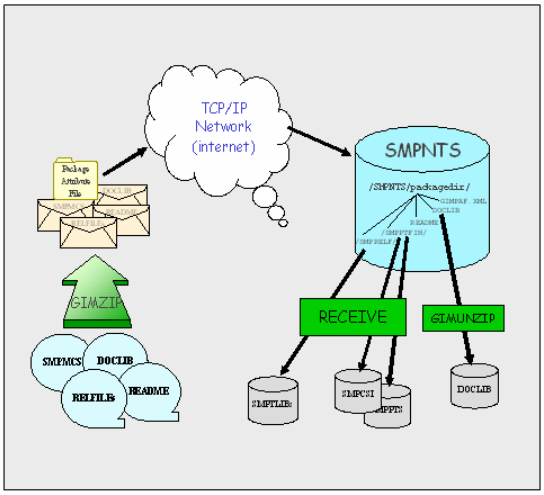
SMP/E provides the GIMZIP and GIMUNZIP service routines to construct, and then later unwrap, network transportable packages of software. This also allows you to create your own packages of SMP/E installable software, and then distribute them within your own enterprise, or to other enterprises. Specifically, the GIMZIP service routine will accept partitioned or sequential data sets as input and will create a network transportable package as output. For more information on the GIMZIP and GIMUNZIP service routines, see SMP/E Reference, SA22-7772.

Once a package is made accessible on the FTP server, you can use the SMP/E RECEIVE command to transfer the package through a TCP/IP network directly into an SMP/E environment. The RECEIVE command has been extended with new DELETEPKG, FROMNETWORK, and FROMNTS operands to process these network transportable packages. For more information on the RECEIVE command changes, see SMP/E Commands, SA22-7771.

New CLIENT and SERVER data sets and SMPDIR and SMPNTS directories have been created to support this new processing. For more information on CLIENT, SERVER, SMPDIR, and SMPNTS, see SMP/E Reference, SA22-7772.

Network RECEIVE Function

- Physically transfers the packages across a TCP/IP network
- Extracts the original data from the packages
- Performs traditional RECEIVE operations on the original data



11

Uses two distinct components of SMP/E:
1) GIMZIP network packaging service routine Creates transportable packages of:

- Modification Control Statements (MCS) RELFILEs
- HOLDDATA
- Any other associated data (doc, samples, etc.)

2) GIMUNZIP service routine to extract the original data from the packages

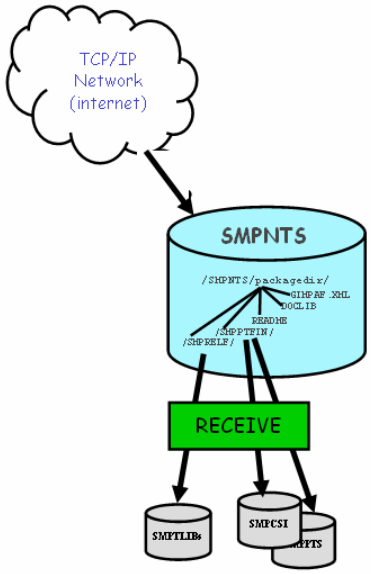
The transfer places the package into directories in the SMPNTS (HFS) data set. Each package has its own directory. For example:
/shared/smpnts/
dir U00076813
dir U00074517

/shared/smpnts/U00074517
File GIMPAF.XML
File GIMPAF.XSL
Dir SMPHOLD ←==== SMP/E HOLDDATA
Dir SMPPTFIN ←==== Points to the source of the SYSMODs
Dir SMPRELF ←==== Product Relfiles

IBM

RECEIVE Command Extensions

- RECEIVE will extract the original data from the package
  - Expand each archive and reconstruct the original data sets
- Perform traditional RECEIVE operations on the original data
  - Create SMPTLIBs from RELFILEs
  - Update SMPCSI and SMPPTS
- RECEIVE command options
  - FROMNETWORK
  - FROMNTS



12

Once the package has been successfully transferred from an FTP server to the client and stored in a subdirectory of the SMPNTS, SMP/E RECEIVE processing can begin.

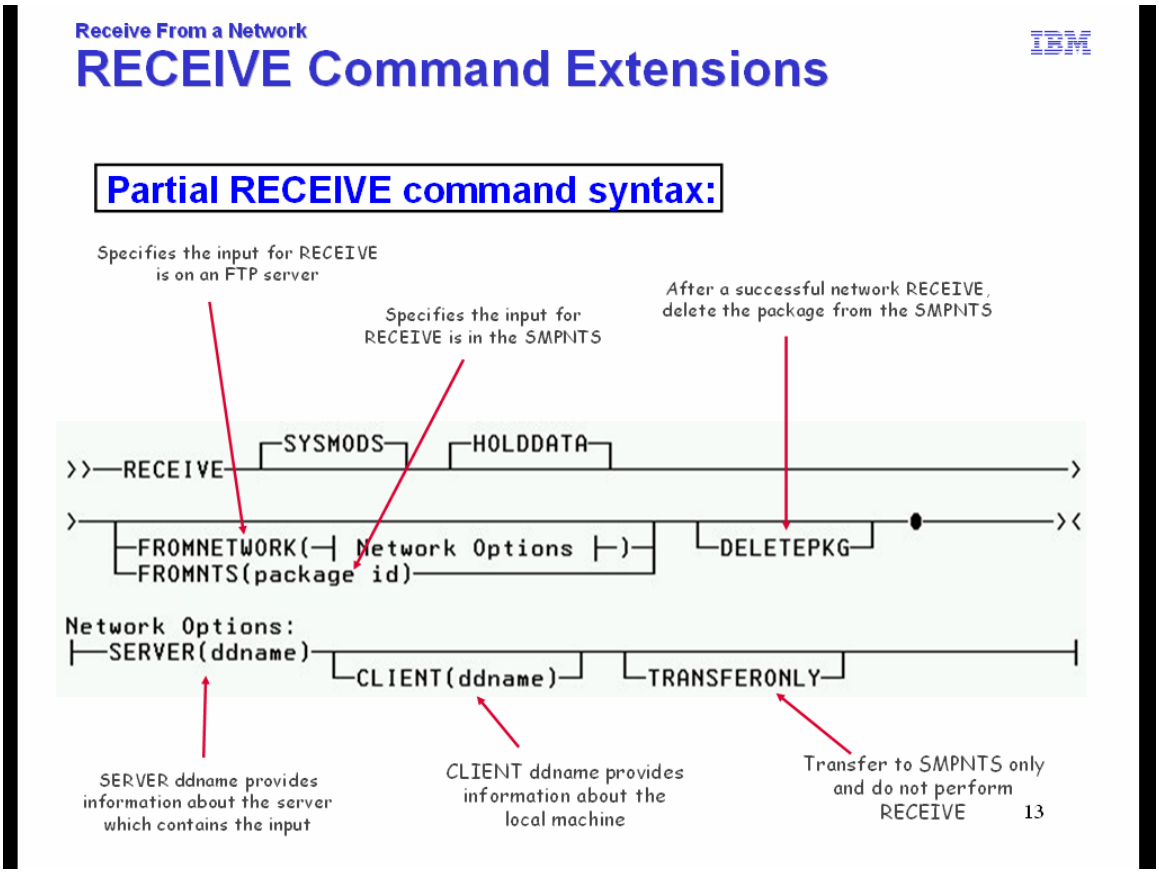
SMP/E reads the package attribute file do determine the files making up the package.

SMP/E transforms the archive files in sub directories /SMPPTFIN, /SMPHOLD and /SMPRELF of the package directory into images of the original data set.

SMP/E will process this data as if it had been read from the SMPPTFIN and SMPHOLD data sets and perform SMP/E RECEIVE processing as directed by the operands on this command.

The SMP/E RECEIVE command options for electronic delivery are FROMNETWORK and FROMNTS. The difference is FROMNETWORK will go through RECEIVE processing whereby the Global zone and the SMPPTS are updated, (business as usual). There is the capability not to actually perform FTP processing by specifying TRANSFEFRONLY with FROMNETWORK. TRANSFERONLY will stop after the package is stored in the SMPNTS, for processing at a later time.

The RECEIVE FROMNTS performs receive processing by retrieving the data in the SMPNTS, as opposed to retrieving it the conventional way, from TAPE or DASD. This is useful when you need to receive the same package into multiple global zones.



FROMNETWORK specifies the input for this RECEIVE command is a GIMZIP package on a TCP/IP connected FTP server. FROMNETWORK can also be specified as FROMNET.

SERVER(ddname) specifies the 1- to 8-character DD or DDDEF name that points to the data set that provides information about the server containing input for this RECEIVE command. SERVER is required if FROMNETWORK is specified.

CLIENT(ddname) specifies the 1- to 8-character DD or DDDEF name pointing to the data set where RECEIVE can get information about the TCP/IP client environment on the local machine.

TRANSFERONLY specifies that RECEIVE FROMNETWORK processing should stop after the network package has been transferred into the SMPNTS file structure of the hierarchical file system.

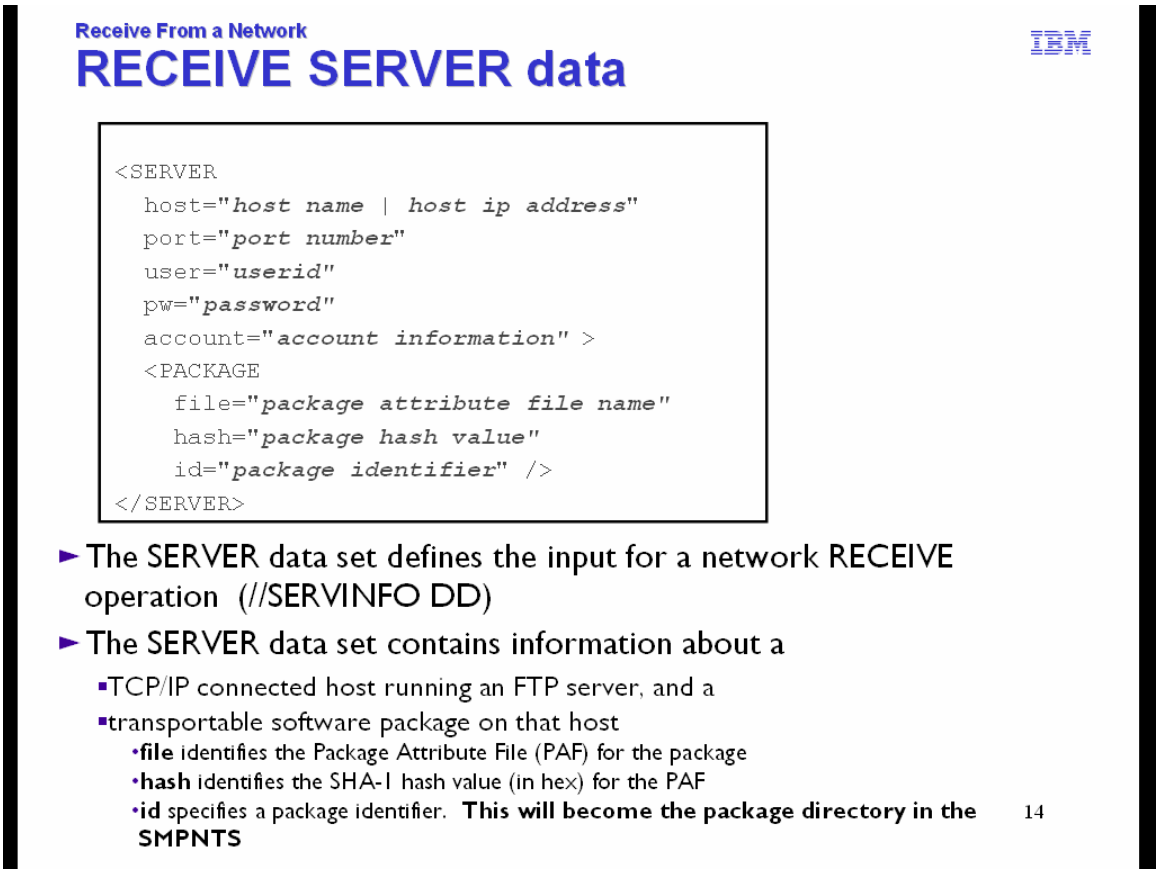
Notes:

1. TRANSFERONLY is mutually exclusive with DELETEPKG.
2. When TRANSFERONLY is specified, the BYPASS, EXCLUDE, FORFMID, HOLDDATA, LIST, SELECT, SOURCEID, SYSMODS, and ZONEGROUP operands do not apply and are not processed if specified.
3. When TRANSFERONLY is not specified, RECEIVE FROMNETWORK processing performs the more typical function of RECEIVE (in addition to transferring the network package) by processing SYSMODs and HOLDDATA from the just transferred network

package into the global zone, the SMPPTS, and temporary data sets (SMPTLIBs) for later SMP/E processing.

Example:

```
//*
//SMPER1  EXEC PGM=GIMSMP,REGION=0M,
//             PARM='PROCESS=WAIT'
//SMPCSI   DD  DISP=SHR,DSN=global_csi_data_set_name
//SMPNTS   DD  PATHDISP=KEEP,
//             PATH='SMPNTS path name'
//SMPOUT   DD  SYSOUT=*
//SMPRPT   DD  SYSOUT=*
//SMPLIST  DD  SYSOUT=*
//SMPLOG   DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//SYSUT1   DD  UNIT=SYSALLDA,SPACE=(6160,(0230,0760))
//SYSUT2   DD  UNIT=SYSALLDA,SPACE=(6160,(0230,0760))
//SYSUT3   DD  UNIT=SYSALLDA,SPACE=(6160,(0230,0760))
//SYSUT4   DD  UNIT=SYSALLDA,SPACE=(6160,(0230,0760))
//SMPCNTL  DD  *
  SET      BOUNDARY (GLOBAL) .
  RECEIVE SYSMODS
          HOLDDATA
          FROMNETWORK(
           SERVER(SERVINFO)
       /*  CLIENT(CLNTINFO) */
       /*  TRANSFERONLY  */
                       ).
/*
//*
//SERVINFO DD *
  <SERVER
    host="inetsd01.boulder.ibm.com"
    user="B723066w"
    pw="b382395a"
    >
    <PACKAGE
      file="2004073008473/PROD/GIMPAF.XML"
      hash="951E2234BFC89556EF0990133CECAF1501D294B7"
      id="U00074517"> <!-- NOTE 5 -->
    </PACKAGE>
  </SERVER>
/*
//CLNTINFO DD *                                <=== NOTE 4
  <CLIENT>
    <FIREWALL>
      <SERVER
        host="local_firewall_server_host"
        user="local_firewall_user"
        pw="local_firewall_pw"
        >
      </SERVER>
      <FIRECMD>USER &REMOTE_USER;@&REMOTE_HOST;</FIRECMD>
      <FIRECMD>PASS &REMOTE_PW;</FIRECMD>
     </FIREWALL>
  </CLIENT>
/*
```



The RECEIVE FROMNETWORK command requires a SERVER data set be defined to provide the necessary information about the FTP server from which a network package is to be received and about the network package itself. A CLIENT data set may also be required to provide information about the local host, such as information about the local firewall requirements.

Content of SERVER Data Set

The SERVER data set contains information about a TCP/IP connected host running an FTP server.

<SERVER> specifies the start of SERVER data. This tag is required.

The attributes for the <SERVER> tag are:

host

identifies the FTP server from which a package is to be received. The host attribute must specify either:

host name a fully qualified internet host name that can be resolved by the domain name system to an internet address. A host name is a text string up to 255 characters drawn from the alphabet (A-Z), digits (0-9), period (.), and minus sign (-). Periods are allowed only as delimiters within domain style names. No blank or space characters are permitted as part of a name. No distinction is made between upper and lower case. The first character must be a letter or a digit. The last character must not be a minus sign or period.

host ip address

an internet address defining the host's location on the internet, specified in dotted decimal notation. The dotted decimal form is x.x.x.x, where each " x" represents 1 to

3 digits (0-9). One host attribute specifying either a host name or a host ip address is required. The host attribute value must be enclosed in quotation marks

user
specifies a userid that will give you access to the host machine. This attribute is a text string up to 80 characters of any type and must be enclosed in quotation marks. If anonymous logins are accepted by this host, specify user=″ anonymous″ . The user attribute is required.

pw
specifies a password value that will give you access to this host. This attribute is a text string up to 80 characters of any type and must be enclosed in quotation marks. If anonymous login is being used, specify your e-mail address as the password value. The pw attribute is required.
Note: Use system facilities to restrict access to the data set named on the SERVER operand to ensure the security of the password value.

account
specifies the account information for the specified userid. This attribute is a text string up to 80 characters of any type and must be enclosed in quotation marks. The account attribute is optional.

port
specifies the port number to be used for TCP/IP operations on the specified host. The port number must be a decimal number in the range 1 through 65535. This attribute value must be enclosed in quotation marks. The port attribute is optional.

</SERVER> specifies the end of SERVER data. This tag is required.

<PACKAGE> specifies the start of PACKAGE data. All data about the package to be retrieved from this SERVER follows this tag and precedes the </PACKAGE> tag. This tag is required. Only one PACKAGE tag may be specified within the SERVER data. The attributes for the <PACKAGE> tag are:

file
specifies the full name and path of a package attribute file. This file contains a list of all the files that are to be transmitted by this RECEIVE command, as well as information about those files. The path can contain 1 to 266 characters of type X'40' through X'FE'. The file attribute value is required and must be enclosed in quotation marks.

hash
specifies the 40 character hexadecimal representation of the 20 byte SHA-1 hash value of the package attribute file. The hash attribute value is required and must be enclosed in quotation marks.

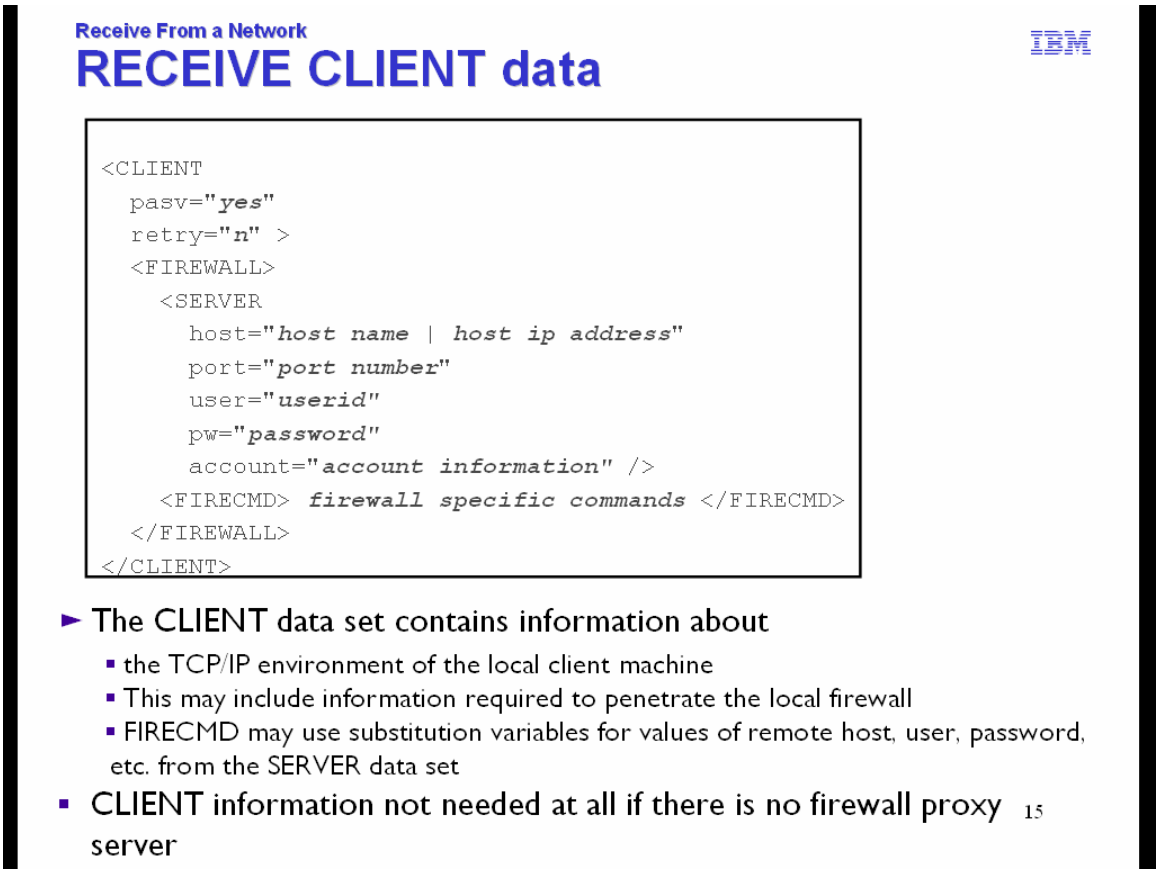
id
specifies a 1 to 50 character value assigned to identify the package by the SMP/E user. This value is used to name the subdirectory within the SMPNTS directory where the transferred files are staged. The value can contain any character from X'41' through X'FE', except a slash (″/″). The id attribute value is required and must be enclosed in quotation marks.

</PACKAGE> specifies the end of PACKAGE data. This tag is required.

Example:

```
//*
//* The information provided in the SERVINFO DD has
//* been customized for your order, please do not
//* alter this information.
//*
```

```
//SERVINFO DD *
  <SERVER
    host="inetsd01.boulder.ibm.com"
    user="B723066w"
    pw="b382395a"
    >
    <PACKAGE
      file="2004073008473/PROD/GIMPAF.XML"
      hash="951E2234BFC89556EF0990133CECAF1501D294B7"
      id="U00074517"> <!-- NOTE 5 -->
    </PACKAGE>
   </SERVER>
```



Content of CLIENT Data Set contains information about the TCP/IP client environment of the local machine.

<CLIENT>

specifies the start of TCP client environment data. This tag is required. The attributes for the <CLIENT> tag are:

retry

One numeric character (0-9) indicating the number of times to retry the transfer of a file when its SHA-1 hash value does not match the hash value expected for this file. This tag is optional. If the retry attribute is not specified, no retries will be attempted. This attribute must be enclosed in quotation marks.

pasv

specifies whether RECEIVE FROMNETWORK processing should attempt to use passive transfer mode with the FTP server for the network package. A value of YES indicates that the usage of passive transfer mode should be attempted. A value of NO is equivalent to not specifying the pasv attribute and indicates the FTP server is to do an active open on a port number provided by SMP/E to transfer the data composing the network package. The attribute value must be enclosed in quotation marks. The value may be entered in mixed case, but is folded to upper case for verification. The pasv attribute is optional.

Following the <CLIENT> tag are:

<FIREWALL>

specifies the start of the firewall section of the TCP client environment data The firewall section of the CLIENT data set should be specified if your TCP/IP environment requires that you connect to the firewall machine to execute FTP commands. Following the <FIREWALL> tag are:

<SERVER>

specifies the start of the server information of the firewall section of the TCP client environment data. This tag is required if <FIREWALL> was specified. The <SERVER> section of the <FIREWALL> entry contains the information SMP/E needs to connect to the firewall host. The attributes for the <SERVER> tag are:

host

identifies the firewall host. The host attribute must specify either:

host name

a fully qualified internet name for the firewall host that can be resolved by the domain name system to an internet address. A host name is a text string up to 255 characters drawn from the alphabet (A-Z), digits (0-9), period (.), and minus sign (-). Periods are allowed only as delimiters within domain style names. No blank or space characters are permitted as part of a name. No distinction is made between upper and lower case. The first character must be a letter or a digit. The last character must not be a minus sign or period.

host ip address

an internet address defining the firewall host's location on the internet, specified in dotted decimal notation. The dotted decimal form is x.x.x.x, where each ″ x″ represents 1 to 3 digits (0-9). One host attribute specifying either a host name or a host ip address is required. The host attribute value must be enclosed in quotation marks

user

id that will give you access to the firewall host machine. This attribute must be enclosed in quotation marks.

pw

specifies a password value that will give you access to this firewall host. This attribute must be enclosed in quotation marks.

Note: Users should use existing system facilities to restrict access to the data set named on the CLIENT operand to ensure the security of the password value.

account

specifies the account information to be used for the specified field is a text string up to 80 characters of any type. The account attribute is optional.

port

specifies the port number to be used for TCP/IP operations on the specified host. The port number must be a decimal number in the range 1 through 65535. This attribute value must be enclosed in quotation marks. The port attribute is optional.

</SERVER>

specifies the end of server section of the TCP client environment data. This tag is required if <SERVER> was specified.

<FIRECMD>

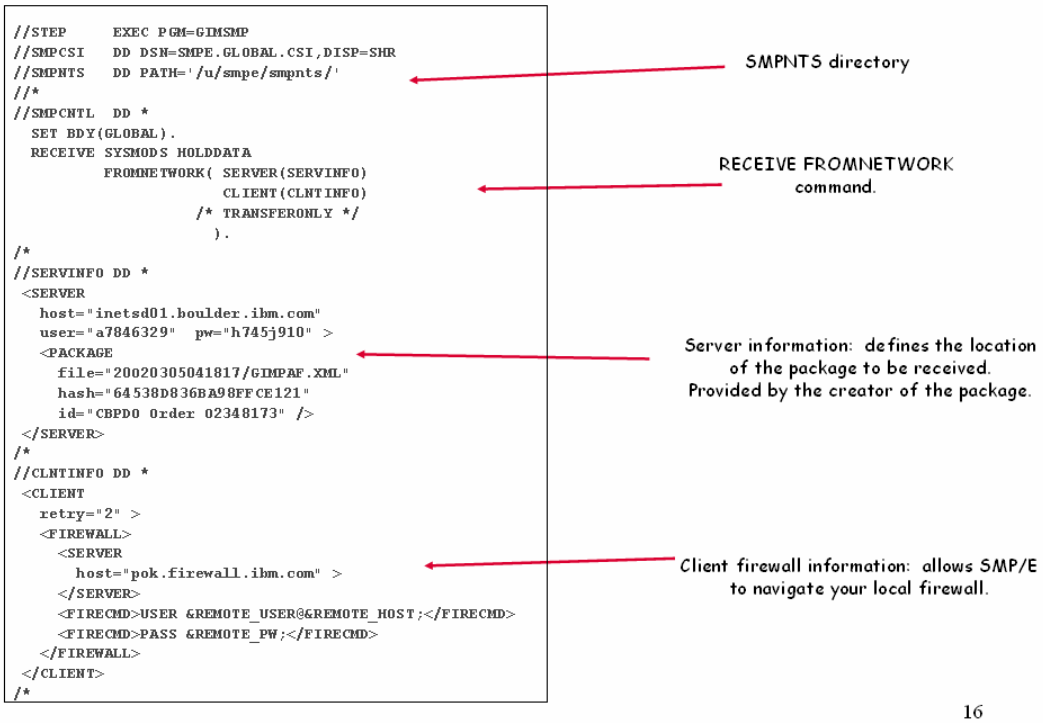
specifies the start of the firewall specific command section of the TCP client environment data. This tag is required if the <FIREWALL> tag is specified. Enter the sequence of commands necessary to FTP to an outside host through your firewall. Each command must be on a separate line in the data set. Each command is a text string of up to 80 characters of any type. in this CLIENT data set, is substituted into the command string. If the user attribute is omitted, blanks will be substituted

</FIRECMD>
specifies the end of firewall specific command section of the TCP client environment data. This tag is required if <FIRECMD> was specified.
</FIREWALL>
specifies the end of firewall section of the TCP client environment data. This tag is required if <FIREWALL> was specified.
</CLIENT>
specifies the end of TCP client environment data. This tag is required.

Example:

```
/*
//CLNTINFO DD *
  <CLIENT>
    <FIREWALL>
      <SERVER
        host="local_firewall_server_host"
        user="local_firewall_user"
        pw="local_firewall_pw"
        >
      </SERVER>
      <FIRECMD>USER &REMOTE_USER;@&REMOTE_HOST;</FIRECMD>
      <FIRECMD>PASS &REMOTE_PW;</FIRECMD>
     </FIREWALL>
  </CLIENT>
/*
```

Sample RECEIVE FROMNETWORK job

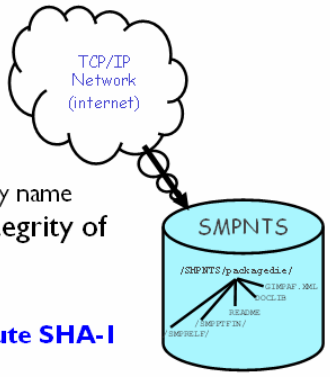


16

RECEIVE FROMNETWORK Processing

Physically transfer a package from a TCP/IP connected server to the client machine:

- The SERVER data set identifies the Package Attribute File (PAF) for the package, therefore it is retrieved
- The PAF describes the archives of the package, therefore they are each retrieved
- The PAF and archives for the package are stored in a subdirectory of the SMPNTS
  - the package identifier (in SERVER data set) is the subdirectory name
- The SHA-1 hash values are used to determine the integrity of the PAF and the archive files
  - If necessary, SMP/E will retry the network operation
  - **Note: Cryptographic Services (ICSF) is used to compute SHA-1 hash values**
- For restartability, SMP/E checks the SMPNTS for what has already been transferred
  - the hash value is used to determine correctness of any existing archives



17

Processing for RECEIVE FROMNETWORK

When processing a RECEIVE FROMNETWORK command to receive a package from a network location, SMP/E performs the following steps:

1. SMP/E reads the information in the SERVER data set and, optionally, the CLIENT data set. The SERVER data set contains information about the FTP server on which resides the package to be received, as well as information about the package itself, such as its location on the server, an SHA-1 hash value for it, and an identifier for the package. The CLIENT data set contains information such as local fire wall navigation details, and whether file transfer operations should be retried if necessary.
2. Using the information found in the SERVER data set, SMP/E opens a socket with the FTP server that contains the package to be received.
3. SMP/E transfers the package attribute file (GIMPAF.XML) for the package and stores it in the package directory of the SMPNTS. The package identifier value found in the SERVER data set is used as the package directory in the SMPNTS.
4. SMP/E computes the SHA-1 hash value for the transferred file. If the computed hash value matches the hash value found in the SERVER data set information, then the file was transferred properly. If the hash values do not match, then the file may have been corrupted during its transfer (another possibility is that the value in the SERVER data set is incorrect). If a nonzero RETRY value was specified in the CLIENT data set information, SMP/E retries the transfer operation. Otherwise, RECEIVE processing will fail.
5. The package attribute file can be thought of as a "packing list" for the package to be received. It contains a list of all the files that, combined, make up the entire package. SMP/E reads the package attribute file (if successfully stored) to determine the files that

make up the package. SMP/E then transfers each file defined in the package attribute file from the FTP server and stores it in the package directory of the SMPNTS. SMP/E creates subdirectories as needed in the package directory and stores files in the subdirectories according to their **filetype** or **subdir** attributes indicated in the package attribute file. SMPPTFIN archive files are stored in the /SMPPTFIN subdirectory, SMPHOLD archive files are stored in the /SMPHOLD subdirectory, and SMPRELF archive files are stored in the /SMPRELF subdirectory. All other files are stored in the subdirectory specified on the **subdir** attribute or, if the **subdir** attribute is not present, in the package directory with no subdirectory.

6. SMP/E computes the SHA-1 hash value for each file it stores. If the computed hash value matches the known hash value indicated in the package attribute file, the file was transferred properly. If the hash values do not match, then the file has been corrupted during its transfer. If a nonzero RETRY value was specified in the CLIENT data set information, SMP/E retries the transfer operation. Otherwise, RECEIVE processing will fail.
7. SMP/E closes the socket with the FTP server after all files of the package have been transferred successfully or when no further retry operations will be performed.
8. SMP/E processes the package contents (unless TRANSFERONLY has been specified on the RECEIVE command) as directed by the applicable RECEIVE command operands. If SYSMODs or HOLDDATA are to be processed, SMP/E transforms the archive files in the /SMPPTFIN, /SMPHOLD, and /SMPRELF subdirectories of the package directory into images of the original data. SMP/E then processes this data as if it had been read from the SMPPTFIN and SMPHOLD data sets.

Restarting RECEIVE FROMNETWORK

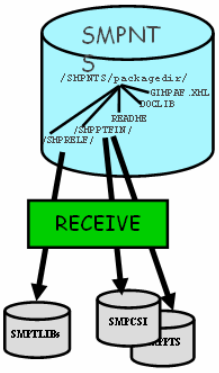
If errors occur during the transfer of files for a package, it may be necessary to restart the RECEIVE FROMNETWORK operation in order to complete the transfer of the entire package. You can simply rerun the RECEIVE FROMNETWORK command, and SMP/E will determine which files have already been transferred successfully, and which files need to be transferred again or have not yet been transferred. Before transferring a file, RECEIVE processing checks to see if the file already exists in the package directory of the SMPNTS. If it does, the hash value for the existing file is calculated and compared to the hash value supplied in the package attribute file. If they match, the file is used as it exists in the package
directory and is not transferred again. If the hash value of the existing file does not match the value supplied in the package attribute file, the file is transferred from the FTP server and replaces the existing file in the package directory.

Notes:
SMP/E V3R3 uses the standard FTP Client instead of using special code.

Receive From a Network

RECEIVE FROMNTS Processing

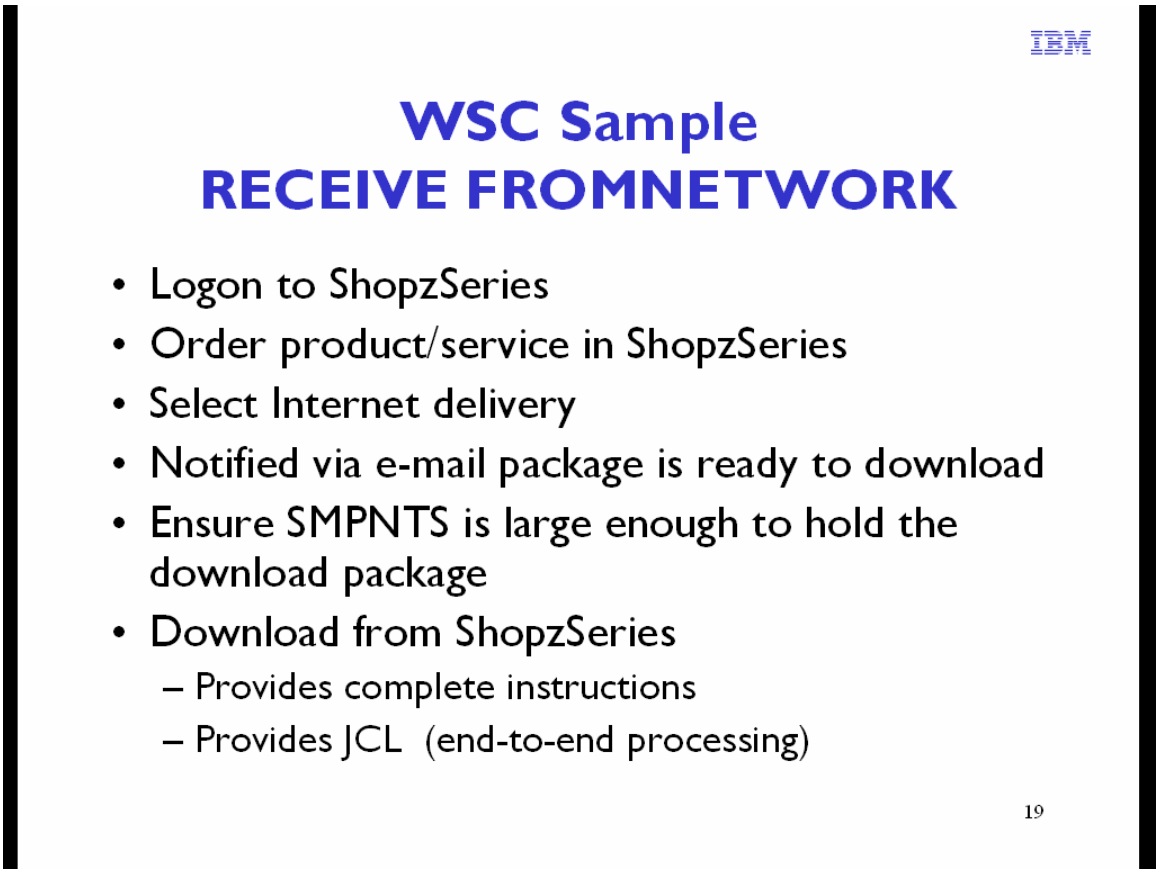
- SMPNTS is a directory in the HFS
  - Network Temporary Store (NTS) for GIMZIP packages
  - somewhat analagous to your tape library
  - specified by DD or DDDEF with ddname SMPNTS
  - for example: /u/smpe/smpnts/
- Extract the original data from the package:
  - the package resides in a subdirectory of the SMPNTS
  - pax is used to uncompress and expand some archives
    - SMPPTFIN, SMPHOLD, and SMPRELF archives only
  - reconstruct the original data using information from the File Attribute File (FAF) in the archive
- Perform traditional RECEIVE operations on the original data:
  - Copy RELFILEs to SMPTLIB data sets
  - Update global zone and SMPPTS from SMPPTFIN and SMPHOLD



18

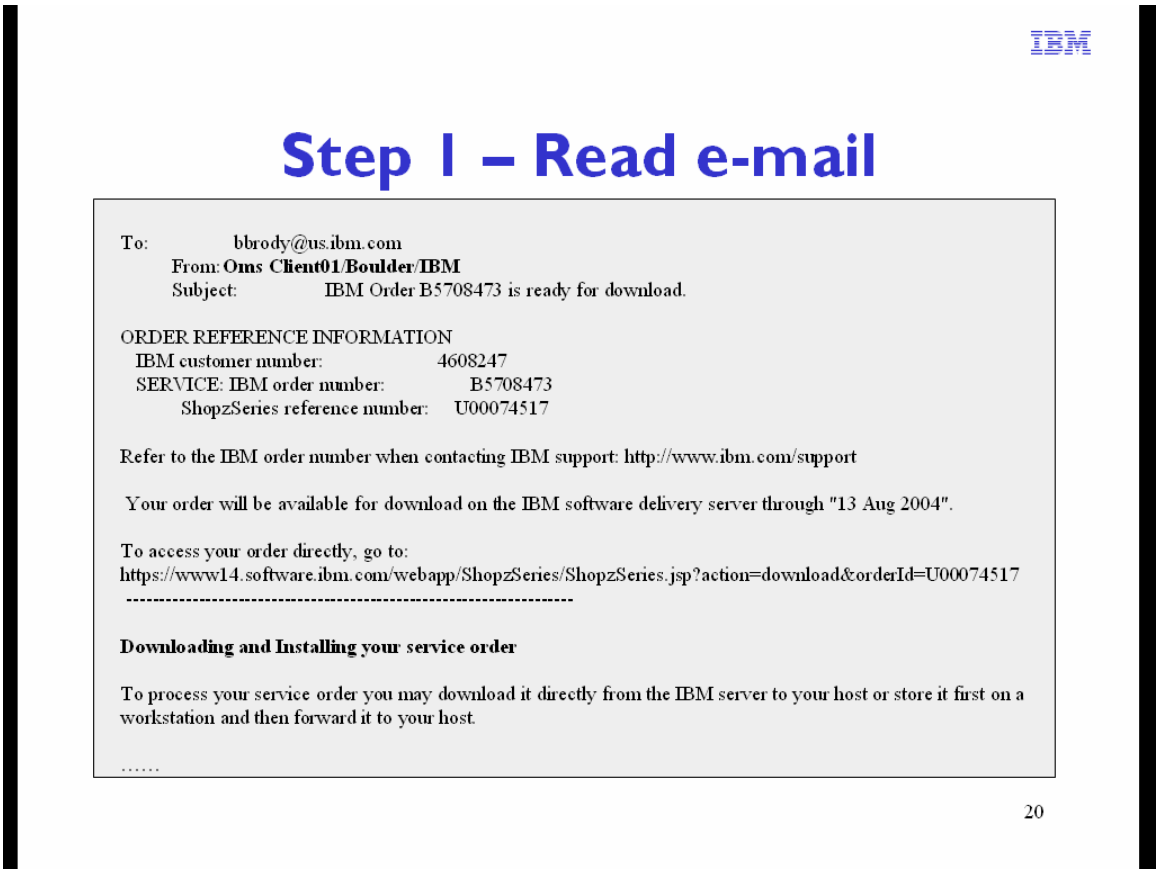
Processing for RECEIVE FROMNTS

When processing a package that already exists in the SMPNTS using the RECEIVE FROMNTS command, SMP/E processes the package contents, as directed by the applicable RECEIVE command operands. If SYSMODs or HOLDDATA are to be processed, then SMP/E transforms the archive files in the /SMPPTFIN, /SMPHOLD, and /SMPRELF subdirectories of the package directory into images of the original data. SMP/E then processes this data as if it had been read from the SMPPTFIN and SMPHOLD data sets.



Internet delivery is only available through ShopzSeries! The package documentation and e-mail contains all JCL to receive it using SMP/E.

You will need to upload this JCL to the host or use copy paste.



Downloading and Installing Your Order

To process your service or product order you may download it directly from the IBM server to your host or store it first on a workstation and then forward it to your host.

A. Download from the IBM server directly to your host using one of the following methods:

1. Use the SMP/E RECEIVE FROMNETWORK command to download your order from the IBM server and stage it to your host's GLOBAL zone. The ShopzSeries order download page contains customized RECEIVE FROMNETWORK JCL for your order. Just click the "Download package directly to host using JCL job" link on the download page and follow the instructions in the JCL comments.

One of the following or higher is required to support RECEIVE FROMNETWORK

- SMP/E version 3.1
- z/OS version 1.2
- z/OS.e

2. Use the sample batch FTP JCL to transfer the order files from the IBM server to your host. Note that FTP does not do data integrity checking. If you use this method, you must ensure the order files have been transferred in their entirety and the contents of the files were not altered during the FTP transfer by specifying the HASH=YES parameter on the GIMUNZIP job. Access the batch FTP JCL by selecting the

"Download package directly to host using FTP" link on your order download page. Once the package files are in your host's HFS, use the sample GIMUNZIP job referenced in section "B.2.a"of this file.

B. Store on workstation then forward to host

1. Download your order to a workstation using one of the following methods:
   a) Use Download Director, a browser-based HTTP transfer program to store your order files on a workstation. Launch the download by selecting the "Download to your workstation using IBM Download Director" link on your order download page. A java applet will download your order. It may be necessary to grant permissions to this applet so that it may write the order files to your workstation.

   b) Use FTP through your browser to download the order files to your workstation. Select the "Download to your workstation using FTP" link on your order download page to invoke FTP. From the screen displayed, download the order files using the facilities provided by your browser. Note that FTP does not do data integrity checking. If you use this method, you must ensure that the order files have been transferred in their entirety and that the contents of the files were not altered during the FTP transfer. You can do this by using GIMUNZIP to process the order files and specifying the HASH=YES parameter.

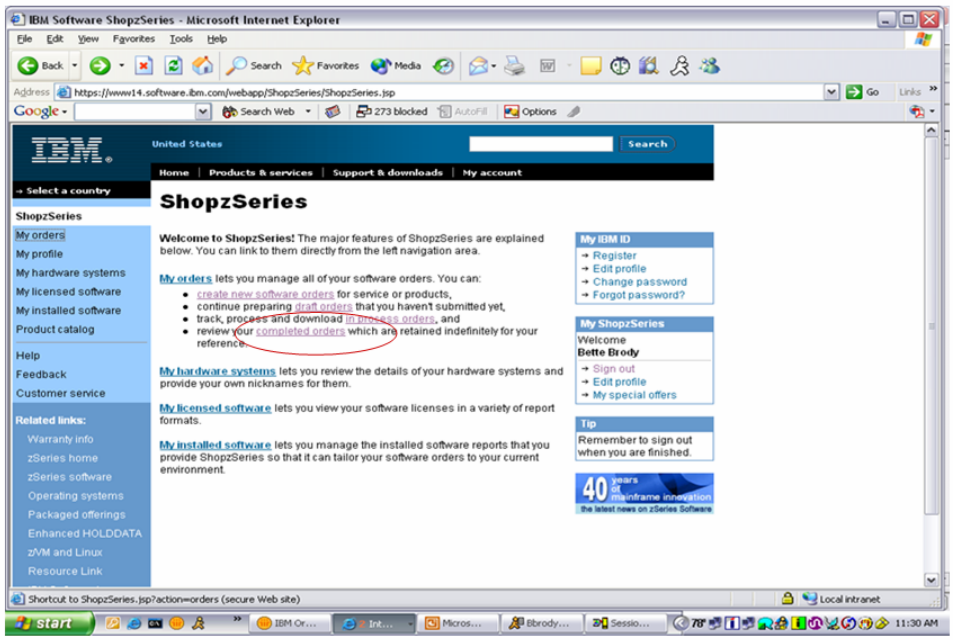
Note that with either of the methods described above you have the ability to specify where the package files are stored on your workstation. If you select a directory on your workstation that you have mapped as a network drive that points to a directory in your host's HFS (using DFS SMB or a similar technology), then these methods have effectively transferred the package files directly to your host.

2. Make the order files accessible to SMP/E to initiate the installation on your host system. There are three methods which can be used to prepare for SMP/E RECEIVE processing:
   a) Use the SMP/E GIMUNZIP utility. This method requires the SMP/E GIMUNZIP utility to access package files from a directory mounted on your host. This directory may be local to your host system or remote. "Local" means that the package files have been manually uploaded from your workstation to the hierachical file system (HFS) on your host. Note the files must be transferred in binary mode and the directory structure must be maintained. Directory and file names are case sensitive. "Remote" means that the package files reside on your workstation in a directory that has been mounted on your host. The directory can be mounted using NFS (Network File System) or a similar technology.

Edit the following sample JCL to expand the files in your order into sequential data sets that can be used as input to SMP/E RECEIVE processing:

Step 2

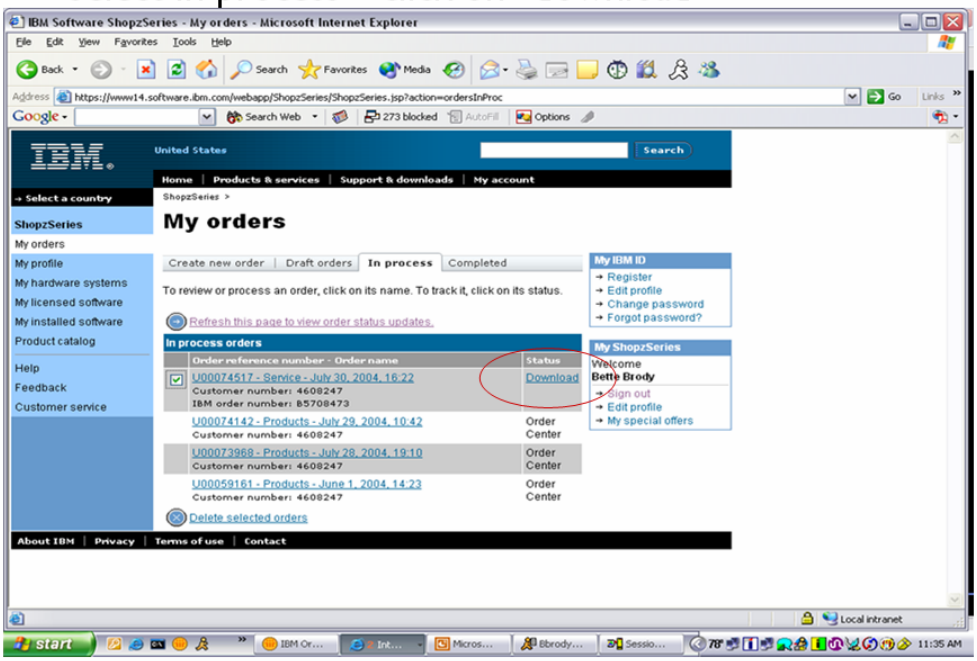
- Go to ShopzSeries web site – completed orders



21

Step 3

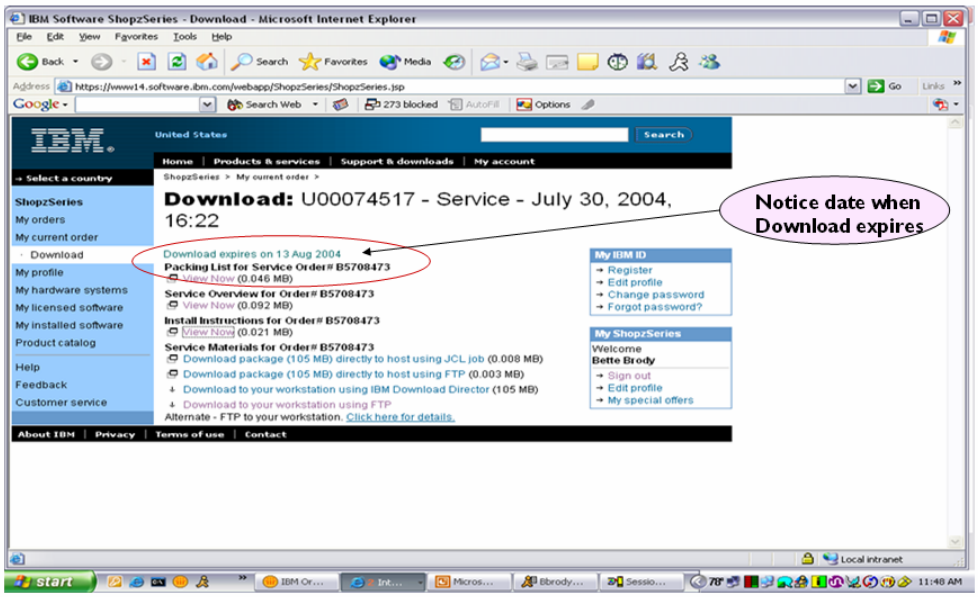
- Select In process – click on "download"



22

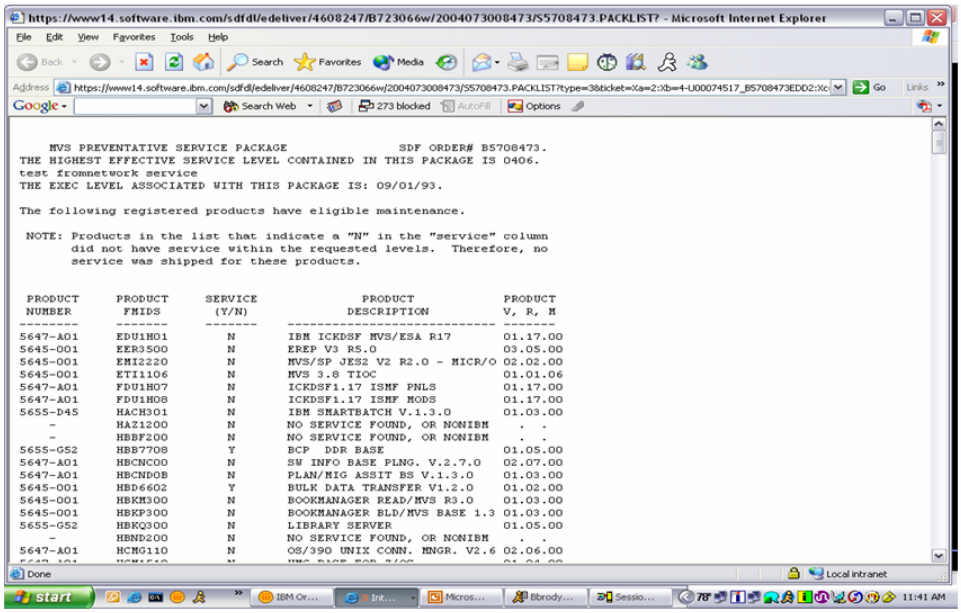
Step 4

- Review "packing list" for order



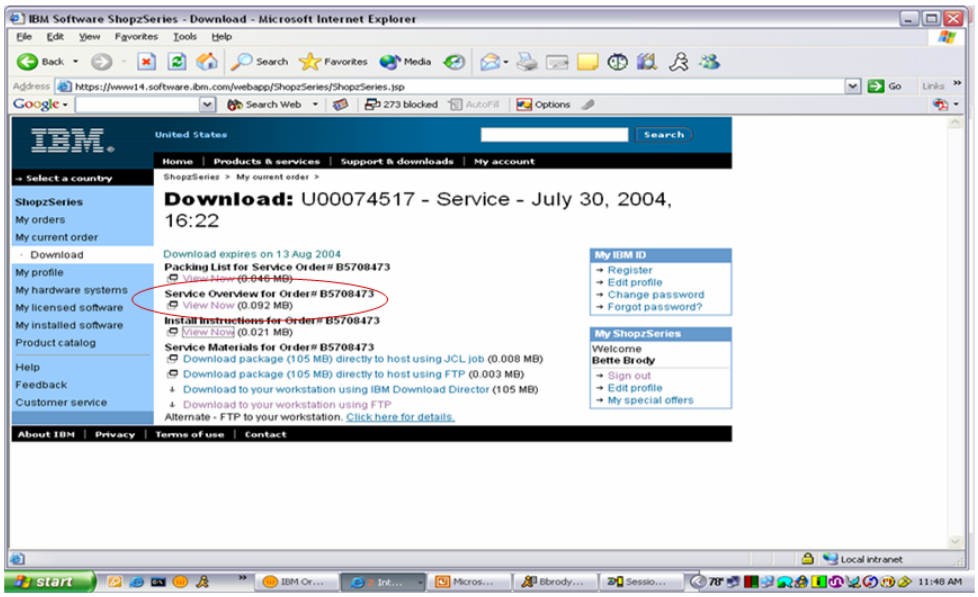
Step 4...

- View Packing List for Order



Step 5

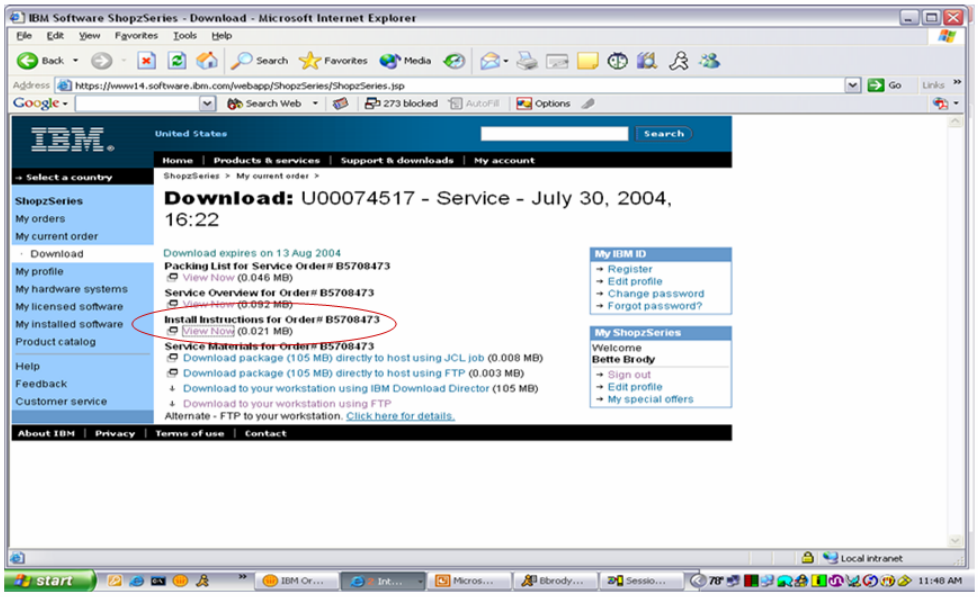
- Review Service Overview for order



25

Step 6

- Review Install Instructions for order



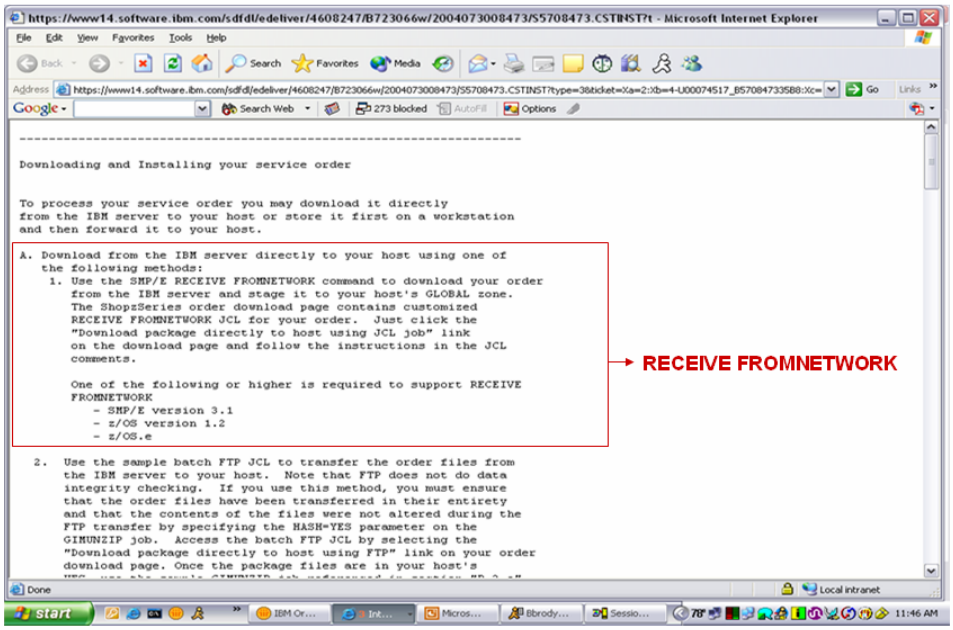
26

You have choices for downloading the package. Installation instructions are provided for:

- Download package directly to host using JCL job (RECEIVE FROMNETWORK)
- Download package directly to host using FTP
- Download to your workstation using IBM Download Director
- Download to your workstation using FTP

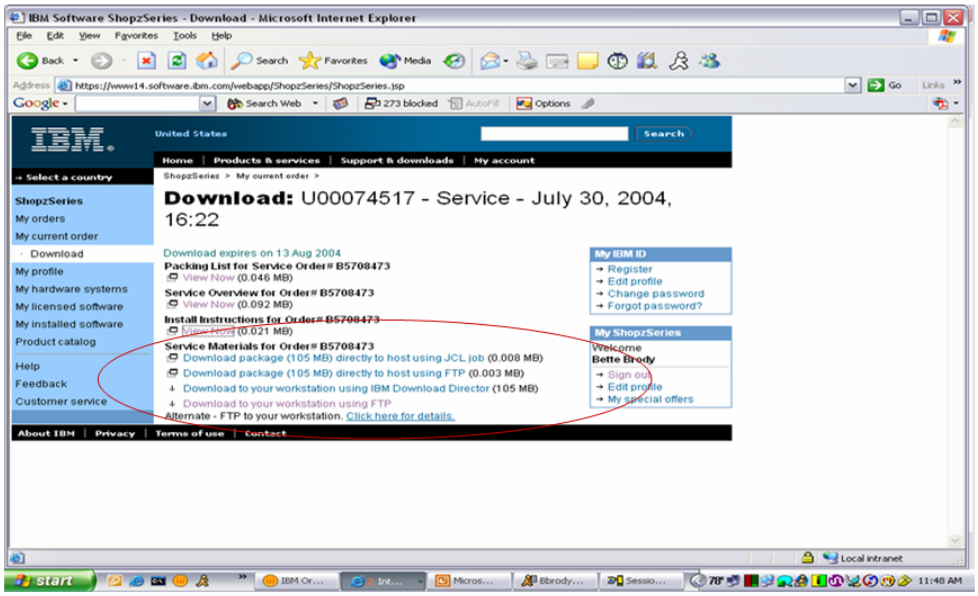
Step 6...

- View Install Instructions for Order



Step 7

- Click on "*your download preference*" (select from list)



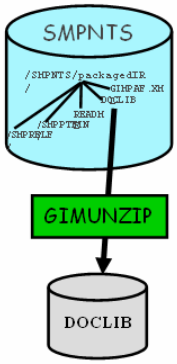
28

Upload and Modify JCL.

GIMUNZIP Service Routine

GIMUNZIP is used to extract the original data from those archives not processed by RECEIVE

```
<GIMUNZIP>
  <ARCHDEF
    name="archive file name"
    volume="volser"
    prefix="data set prefix"
    newname="data set name" />
</GIMUNZIP>
```



- The ARCHDEF statements define the input archives for GIMUNZIP
  - **name** identifies the archive file
  - **volume** identifies the VOLSER to use for the output data set
  - **prefix** identifies the data set prefix for the output data set
  - **newname** provides a new name for the output data set
- Each ARCHDEF defines a single archive file to be expanded into its original data set
  - pax is used to uncompress and expand the archive
  - reconstruct the original data using the information from the File Attribute File (FAF) in the archive

29

GIMUNZIP is a SMP/E service routine used to extract files contained in network transportable packages that were built using GIMZIP.

The GIMUNZIP service routine is used to extract data sets from archive files in GIMZIP packages created by the GIMZIP service routine. These packages typically contain software and associated materials in the form of SYSMODs, RELFILE data sets, HOLDDATA, and other materials such as documentation, samples, and text files. These GIMZIP packages may be transported through a network, processed by the GIMUNZIP service routine, and then processed by the SMP/E RECEIVE command.

More specifically, the GIMUNZIP service routine extracts data sets from the archive files that compose the GIMZIP package. An archive file consists of a portable image of a sequential or partitioned data set and the information needed to create a data set from the portable image.

Notes:

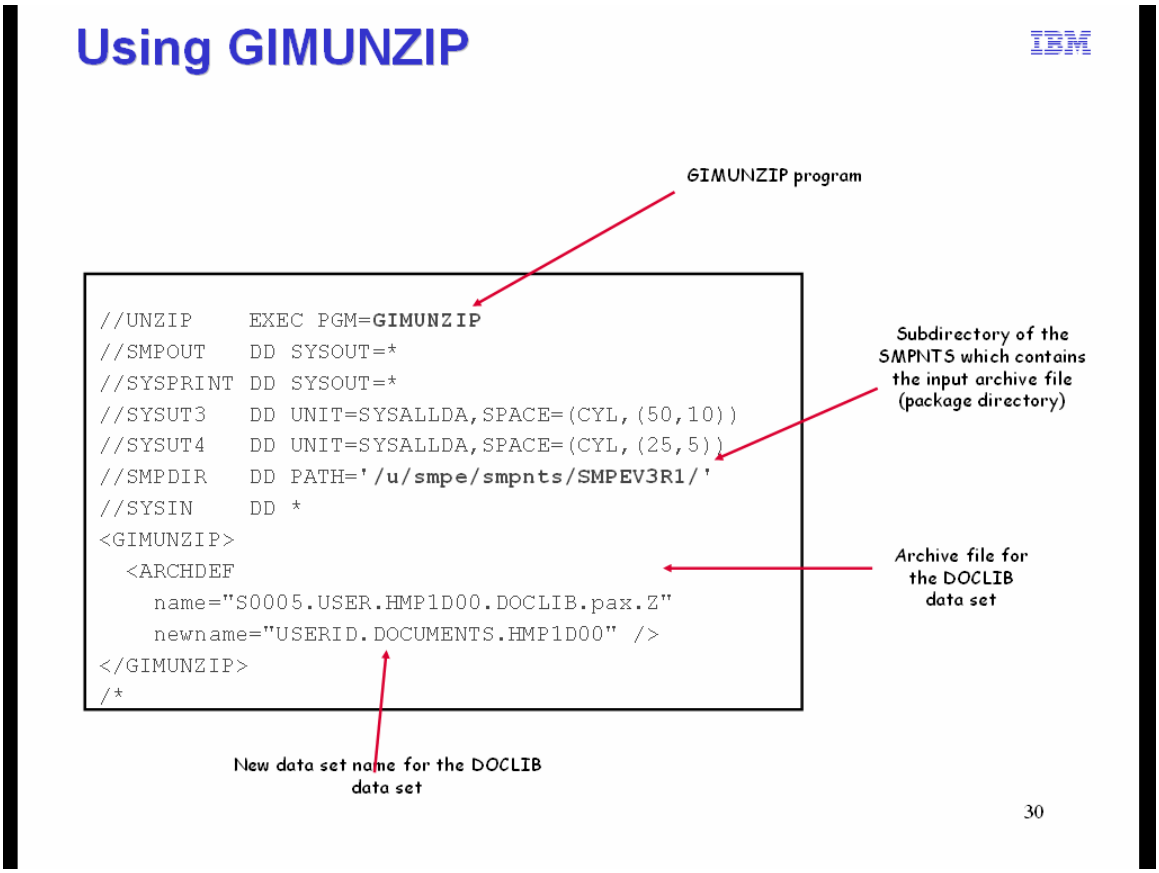
GIMUNZIP is a separate load module residing in the MIGLIB library and runs independently from the rest of SMP/E processing. GIMUNZIP optionally requires the Integrated Cryptographic Services Facility (ICSF) One-Way Hash Generate callable service to be available for its use in order to compute an SHA-1 hash value. The GIMUNZIP service routine extracts data sets from the archive files that compose the GIMZIP package. An archive file consists of a portable image of a sequential or partitioned data set and the information needed to reload the data from the portable image. GIMUNZIP uses the UNIX System Services pax command to expand the following component files from an archive file temporarily into the hierarchical file system:

1. the portable image of the original data set, and
2. the file attribute file that contains the information necessary to reload the data from the archived data set.

GIMUNZIP then reads the file attribute file and uses the information recorded there, along with information specified on the <ARCHDEF> tag, such as volume, newname, or prefix, to allocate a new data set. The portable image of the original data set is stored in this new data set.

If the HASH=YES option was specified, then GIMUNZIP performs SHA-1 hash checking for the archive files. Specifically, GIMUNZIP reads the package attribute file for the package. The package attribute file, GIMPAF.XML, is found in the package directory, and contains the known hash values for the archives. These hash values were recorded by GIMZIP when the package and archives were created. GIMUNZIP then uses ICSF services to compute the current hash values for the archive files, and compares this value to the known hash values for the archive files.

If the computed hash value matches the known hash value, then the integrity of the archive file is ensured. However, if the computed hash value does not match the known hash value, GIMUNZIP stops processing for the archive file. This condition indicates the archive file has been corrupted since it was produced by GIMZIP when the package was created.



After GIMUNZIP has been run, the SMP/E RECEIVE command can be used to process the data sets extracted by GIMUNZIP.

<GIMUNZIP> Tag Syntax
The <GIMUNZIP> and corresponding </GIMUNZIP> tags identify the beginning and end of a set of archive retrieval requests.
<ARCHDEF> Tag Syntax
The <ARCHDEF> and corresponding </ARCHDEF> tags identify the beginning and end of a specific archive retrieval request. The following attributes may be found on the <ARCHDEF> tag:

name=″ archive name″
specifies the path name for an archive file to be extracted by GIMUNZIP. The path name value is a relative value and it is relative to the package directory specified on the SMPDIR DD statement.
volume=″ data set volume″
specifies the volume serial number of the volume on which GIMUNZIP will allocate the data set to be extracted from the archive file. The volume identifier must be from 1 to 6 alphanumeric characters. If no volume is specified, SYSALLDA is used as the unit type.
newname=″ data set name″
specifies the data set name to use for the data set to be extracted from the archive file. This name replaces the original name of the data set as recorded in the archive's file attribute file. The specified data set name can be up to

44-characters long and must conform to standard data set naming conventions. If neither the newname nor the prefix attributes are specified, the original name for the data set is used.

prefix=″ data set prefix″

specifies a data set prefix to use for the data set to be extracted from the archive file. This prefix replaces the high-level qualifier of the original name for the data set as recorded in the archive's file attribute file. The specified data set prefix can be up to 26-characters long and must conform to standard data set naming conventions. If neither the prefix nor the newname attributes are specified, the original name for the data set is used.

Syntax Notes

GIMUNZIP ignores columns 73 through 80. If data is specified beyond column 72, GIMUNZIP ignores it, which may lead to the diagnosis of an error in a following tag. Package control tags may contain comments. Comments start with <!-- (hex 4C5A6060) and end with --> (hex 60606E). The first --> encountered after the initial <!-- will end the comment. A comment may precede or follow a tag, but may not appear within a tag.

Time for...