z/OS V1R13

BCP program binder:
Specifying unnamed sections and symbols

Overview

- Problem Statement / Need Addressed
  - An "unnamed section" or symbol is one that has a binder generated name in the format "$PRIVxxxxxx". These can be seen in:
    - binder SYSPRINT MAP / XREF / messages
    - AMBLIST LISTLOAD output
  - These names are freshly generated on each bind to ensure that each binder-generated name has a unique number, even if several modules with binder-generated names are being linked together
  - They represent binder-generated internal names which are 4 byte binary integers
    - xxxxxx is the hexadecimal representation of those numbers
    - the binder owns numbers through X'00000F'
  - It has always been possible to refer to these binary symbol names using the Binder APIs
    - The length is 4
    - The name is the actual binary value
    - The binder C APIs require the use of __iew_api_name_to_str() so that the name can be passed as a string
  - No mechanism existed for batch invocations

  - Issue originally documented in Marketing Requirement MR0601056745
    - We have done an AMBLIST and have determined that a CONTROL SECTION $PRIV000010 … contains the unresolved reference
    - We have tried to remove this csect, but have not been successful. The client uses JCL that has other steps taking place after the binder and these are now not executing as the bind step has a non zero return code
    - The customer would like IBM to design an easier option than having to re-compile
- Solution
  - Enhance existing Binder option STRIPSEC
    - Add PRIV sub option
      - NO - default
      - YES - removes all unreferenced sections
      - PRIV - subset of YES applying to only unnamed symbols
    - This satisfies the Marketing Requirement… but seems incomplete
  - Recognition of $PRIVxxxxxx names on control statements
    - CHANGE
    - REPLACE
  - Users can now rename or delete binder-generated names which represent user symbols
    - batch interface solution!
    - Can only be "oldname" (user cannot create a $PRIVxxxxxx name)
    - Binder has usurped these names
      - If there were any symbols truly named $PRIVxxxxxx they can no longer be referred to
  - Must be sure the $PRIVxxxxxx name matches binder internal usage
    - **Must rebind** single load module or program object
      - It is insufficient to bind together once from original inputs
      - Will not reliably match binder internal names!
    - Binder SYSPRINT output (or AMBLIST LISTLOAD) against the rebound program module can then be used to determine reliable $PRIVxxxxxx names
    - Rebind again specifying $PRIVxxxxxx names on CHANGE and/or REPLACE control statements
- Benefit / Value
  - Ability to delete unreferenced unnamed sections with new STRIPSEC=PRIV sub option
  - Ability to rename or delete $PRIVxxxxxx symbols at bind time, using CHANGE or REPLACE control statements

Usage and invocation

- Example of new STRIPSEC sub option:

```
//BIND1    EXEC PGM=LINKEDIT,
//         PARM=('STRIPSEC=PRIV,LIST=ALL,MAP,LET,XREF')
```

- Example of new control statement support:
  - Remember to rebind your application first, to find the correct $PRIVxxxxxx name! Then, you can rename it like this:

```
//BIND2    EXEC PGM=LINKEDIT,PARM=('LIST=ALL,MAP,LET,XREF')
//SYSLMOD        DD  DSN=&&PDSELIB,DISP=(OLD,PASS)
//LIB1           DD  DSN=&&OBJLIB,DISP=(OLD,PASS)
//SYSPRINT       DD  SYSOUT=*
//SYSLIN         DD  *
 REPLACE $PRIV000010(NEWNAME)
 INCLUDE LIB1(B695RP1)
```

```
      NAME PROGRAM(R)
    /*
```

Interactions and dependencies

- None.

Migration and coexistence considerations

- None.

Installation

- None.

Session summary

- With this line item, you can now:
  - Use the STRIPSEC=PRIV option to remove all unreferenced, unnamed sections.
  - Specify $PRIVxxxxxx names in Binder CHANGE and REPLACE control statements

Appendix - References

- Publications:
  - SA22-7643-11 z/OS V1R13.0 MVS Program Management: User's Guide and Reference
  - SA22-7644-13 z/OS V1R13.0 MVS Program Management: Advanced Facilities