Fault Analyzer

Musterlösungen zu den Übungen

11. August 2012

Eine Ausarbeitung von:

cps4it
Ralf Seidler • Stromberger Straße 36A • 55411 Bingen
Fon: 06721-992611 • Fax: 06721-992613 • Mail: ralf.seidler@cps4it.de
Internet : http://www.cps4it.de
Steuernummer: 08/220/2497/3, Finanzamt Bingen, Ust-ID : DE214792185

Inhaltsverzeichnis

1 Allgemeines

Zum Schmunzeln

"Glück ist: zu begreifen, wie alles zusammenhängt."

Sten Nadolny (*1942), dt. Schriftsteller

Compileliste TES39: u-id.DUMP.COMPILE.TES39
Compileliste TES46: u-id.DUMP.COMPILE.TES46
Compileliste TES47: u-id.DUMP.COMPILE.TES47

Voraussetzungen und Regeln

- Ruhe für die Suche nach der Abbruchursache ist notwendig im wahrsten Sinn des Wortes. Insbesondere sind wichtig:
  - Telefon umleiten oder abstellen
  - mit Führungskräften Zeitpunkte vereinbaren, wann ein Status gegeben wird (und Chef/in aus dem Zimmer werfen) ;-)
- Umwandlungsdatum für jedes zu analysierende Programm ***immer*** kontrollieren. In der Testumgebung ist sogar oft die Umwandlungsuhrzeit wichtig. *Dieser Punkt ist in den Musterlösungen nicht erwähnt.*
- Die Informationen von Fault Analyzer aufmerksam lesen. Jeder Hinweis kann interessant sein und den nachfolgenden Aufwand drastisch reduzieren.
- Jede gefundene Information prüfen, ob sie plausibel ist. Es kann beispielsweise niemals ein „SET ... TO TRUE“ oder ein „CALL“ zu einem S0C7 führen.
- Alle ~~wesentlichen~~ Schritte und Informationen notieren.
- Die Musterlösungen basieren auf den Ergebnissen, die durch eine Batch Reanalysis erzeugt worden sind. Bei der Fehleranalyse auf Basis der ISPF-Online-Oberfläche können die Informationen minimal anders aussehen.
- Bei einigen Musterlösungen geht es _nicht_ darum, den Grund des Abbruchs zu suchen (das ist: Wo kommt der Mist her, sondern nur die Abbruchstelle mit den Dateninhalten zu finden.

2 S0C7 – 1

Aufgabe

Suchen des abgebrochenen Befehls und der Inhalte aller betroffenen Felder

Dump

u-id. DUMP.JOBLOG.S0C7#01.txt

Musterlösung 1 (ohne Source-Code-Unterstützung)

wichtigste Informationen aus Dump:

```
H1> I B M   F A U L T   A N A L Y Z E R   S Y N O P S I S

A system abend 0C7 occurred in module TES47 program TES47 at offset X'712'.
A program-interruption code 0007 (Data Exception) is associated with this abend
and indicates that:
 A decimal digit or sign was invalid.
The abend was caused by machine instruction FD94D150D160 (DP).
Recently referenced data items:

 The failing operand at 0001D4C8 is the result of a PACK instruction using the
 following zoned decimal data item which contains invalid data:

 Data Item . . . . . : BLL=0003+032
   At Address. . . . . : 3691108A
   Length. . . . . : X'9'
   Data Item Storage . . . : F0F0F0F0 F0F04C4C 4C  *000000<<<*

 Data Item . . . . . : BLL=0003+03C
   At Address. . . . . : 36911094
   Length. . . . . : X'9'
   Data Item Storage . . . : F0F0F0F0 F0F0F0F6 F6  *000000066*
```

suchen Abbruchstelle in Compileliste (von TES47)

find 0712 (wenn mit LIST umgewandelt)

➔ Befehl wird angezeigt

find hexloc (wenn mit NOLIST umgewandelt)

suchen Befehl mit Offset <= 0712

➔ `000102 0006FA COMPUTE` (also Zeile 102 beinhaltet den Befehl)

➔ `000102  00101  COMPUTE FELD-ERGEBNIS-3 = FELD-5 / FELD-6  CL**5 37 59 61`

An welcher Stelle sind die Felder im Dump zu suchen ...

```
000037        00036        10 FELD-ERGEBNIS-3        PIC S9(009) BINARY.          CL*31 BLW=00000+044,0000044 4C
000059        00058        10 FELD-5                 PIC  9(009).                 CL**2 BLL=00003+032,0000032 9C
000061        00060        10 FELD-6                 PIC  9(009).                 CL**2 BLL=00003+03C,000003C 9C
```

Suchen im Dump nach den Inhalten der Felder

```
Program and Entry Point Name: TES47
  At Address. . . . . . . . . : 3691C7E8 (Module TES47 offset X'0')
  Program Length. . . . . . . : X'BE4'
  Program Language. . . . . . : COBOL (Compiled using IBM Enterprise COBOL for
                                z/OS and OS/390 V4 R1 M0 on 2012/05/24 at
                                17:11:27)
...
        Event 3 Program TES47 BLW=0000 (Address 369110F8)
        Event 3 Program TES47 GPR 4 (Address 369110F8)
69110F8                              E3C5E2F4 F7404040 *        TES47   *
6911100        +8  E3C5E2F6 F6404040 00000000 00000000 *TES66   ........*
6911110       +18  00000000 00000000 00000000 00000000 *................*
6911120       +28  00000000 00000000 00000000 F0F0F0F0 *............0000*
6911130       +38  F0F0F0F3 C3000000 00088C00 00000000 *0003C...........*
6911140       +48  00000000 00000000 00000000 00000000 *................*
        Lines 36911150-369111A0 (X'60' bytes) same as above
69111B0       +B8  00000000 00000000 00000000 0000     *.............  *
```


cps4it

```
        Event 3 Program TES47 BLL=0003 (Address 36911058)
        Event 3 Program TES47 GPR 3 (Address 36911058)
6911058                          F0F0F0F0 F0F0F0F0 *        00000000*
6911060        +8  F14FF0F0 F0F0F0F0 F0F1F14F F0F0F0F0 *1|000000011|0000*
6911070       +18  F0F0F0F2 F24FF0F0 F0F0F0F0 F0F0F24F *00022|000000002|*
6911080       +28  F0F0F0F0 F0F0F0F4 F44FF0F0 F0F0F0F0 *000000044|000000*
6911090       +38  4C4C4C4F F0F0F0F0 F0F0F0F6 F64FF0F0 *<<<|000000066|00*
69110A0       +48  40404040 40404040 00000000 00000000 *        ........*
69110B0       +58  00000000 00000000 00000000 00000000 *................*
69110C0       +68  00000000 00000000 00000000 00000000 *................*
69110D0       +78  00000000 00000000 36911000 000000F8 *........j.....8*
69110E0       +88  000000EC 00000000 00000000 00000000 *................*
69110F0       +98  00000000 00000000                   *........        *
```

Feld FELD-ERGEBNIS-3 beinhaltet also Low-Values (klar, weil es das Zielfeld ist und vorher noch nicht benutzt wurde)
Feld FELD-5 beinhaltet 000000<<<
Feld FELD-5 beinhaltet 000000066

Den Inhalt des Feldes FELD-5 hat Fault Analyzer schon als fehlerhaft angezeigt („wichtigste Informationen aus Dump“).

Das alles finden wir also recht schnell. Die weiteren Schritte wären, den Grund zu finden, warum Feld-5 fehlerhaft ist. Dies wird hier nicht betrachtet.

Musterlösung 2 (mit Source-Code-Unterstützung)

wichtigste Informationen aus Dump:

```
- <H1> I B M   F A U L T   A N A L Y Z E R   S Y N O P S I S


A system abend 0C7 occurred in module TES47 program TES47 at offset X'A3A'.

A program-interruption code 0007 (Data Exception) is associated with this abend
and indicates that:

  A decimal digit or sign was invalid.

The cause of the failure was program TES47 in module TES47.  The COBOL source
code that immediately preceded the failure was:

  Source
  Line #
  ------
  000103 00102      COMPUTE FELD-ERGEBNIS-3 = FELD-5 / FELD-6

The COBOL source code for data fields involved in the failure:

  Source
  Line #
  ------
  000037 00036         10 FELD-ERGEBNIS-3          PIC S9(009) BINARY.
  000061 00060         10 FELD-5                   PIC  9(009).
  000063 00062         10 FELD-6                   PIC  9(009).

Data field values at time of abend:

  FELD-ERGEBNIS-3 = 0
  FELD-5          = X'F0F0F0F0F0F04C4C4C'  *** Cause of error ***
  FELD-6          = 66
```

Wow. Da haben wir ja schon alles!!

3 S0C7 – 2

Aufgabe

Suchen des abgebrochenen Befehls und der Inhalte aller betroffenen Felder

Dump

u-id. DUMP.JOBLOG.S0C7#02.txt

Musterlösung (ohne Source-Code-Unterstützung)

wichtigste Informationen aus Dump:

```
<H1> I B M   F A U L T   A N A L Y Z E R   S Y N O P S I S

A system abend 0C7 occurred in module TES47 program TES47 at offset X'6B2'.

A program-interruption code 0007 (Data Exception) is associated with this abend
and indicates that:

  A decimal digit or sign was invalid.

The abend was caused by machine instruction FA44D150D158 (ADD DECIMAL).

Recently referenced data items:
  The failing operand at 0001D4C8 is the result of a PACK instruction using the
  following zoned decimal data item which contains invalid data:

  Data Item . . . . . : BLL=0003+00A
    At Address. . . . . : 36911062
    Length. . . . . : X'9'
    Data Item Storage . . . : F0F0F05B F0F0F0F0 F0  *000$00000*

  Data Item . . . . . : BLL=0003+014
    At Address. . . . . : 3691106C
    Length. . . . . : X'9'
    Data Item Storage . . . : F0F0F0F0 F0F0F0F2 F2  *000000022*
```

suchen Abbruchstelle in Compileliste (von TES47)

find 06B2 (wenn mit LIST umgewandelt)

➔ Befehl wird angezeigt

find hexloc (wenn mit NOLIST umgewandelt)

suchen Befehl mit Offset <= 06B2

➔ `000098 00069A COMPUTE` (also Zeile 98 beinhaltet den Befehl)

➔ `000098   00097      COMPUTE FELD-ERGEBNIS-1 = FELD-1 + FELD-2`

An welcher Stelle sind die Felder im Dump zu suchen ...

```
000051  00050   10 FELD-1           PIC  9(009).     CL**2 BLL=00003+00A,000000A 9C
000053  00052   10 FELD-2           PIC  9(009).     CL**2 BLL=00003+014,0000014 9C
000033  00032   10 FELD-ERGEBNIS-1  PIC S9(009).     CL*30 BLW=00000+034,0000034 9C
```

Suchen im Dump nach den Inhalten der Felder

```
Program and Entry Point Name: TES47
  At Address. . . . . . . . . : 3691C7E8 (Module TES47 offset X'0')
  Program Length. . . . . . . : X'BE4'
  Program Language. . . . . . : COBOL (Compiled using IBM Enterprise COBOL for
                                z/OS and OS/390 V4 R1 M0 on 2012/05/24 at
                                17:11:27)...
<H3> Hex-Dumped Storage

        Event 3 Program TES47 BLW=0000 (Address 369110F8)
69110F8                             E3C5E2F4 F7404040 *        TES47   *
6911100        +8  E3C5E2F6 F6404040 00000000 00000000 *TES66   ........*
6911110       +18  00000000 00000000 00000000 00000000 *................*
        Lines 36911120-369111A0 (X'90' bytes) same as above
69111B0       +B8  00000000 00000000 00000000 0000     *............    *

...

        Event 3 Program TES47 BLL=0003 (Address 36911058)
        Event 3 Program TES47 GPR 3 (Address 36911058)
6911058                             F0F0F0F0 F0F0F0F0 *        00000000*
6911060        +8  F14FF0F0 F05BF0F0 F0F0F04F F0F0F0F0 *1|000$00000|0000*
```


cps4it

```
6911070       +18  F0F0F0F2 F24FF0F0 F0F0F0F0 F0F0F24F *00022|000000002|*
6911080       +28  F0F0F0F0 F0F0F0F4 F44FF0F0 F0F0F0F0 *000000044|000000*
6911090       +38  4C4C4C4F F0F0F0F0 F0F0F0F6 F64FF0F0 *<<<|000000066|00*
69110A0       +48  40404040 40404040 00000000 00000000 *        ........*
69110B0       +58  00000000 00000000 00000000 00000000 *................*
69110C0       +68  00000000 00000000 00000000 00000000 *................*
69110D0       +78  00000000 00000000 36911000 000000F8 *........j.....8*
69110E0       +88  000000EC 00000000 00000000 00000000 *................*
69110F0       +98  00000000 00000000
```

Feld FELD-ERGEBNIS-1 beinhaltet also Low-Values (klar, weil es das Zielfeld ist)
Feld FELD-1 beinhaltet 000$00000
Feld FELD-2 beinhaltet 000000022

Den Inhalt des Feldes FELD-1 hat Fault Analyzer schon als fehlerhaft angezeigt („wichtigste Informationen aus Dump“).

4 S0C7 – 3

Aufgabe

Suchen des abgebrochenen Befehls und der Inhalte aller betroffenen Felder
Anschließend: „Umfeld“ des abgebrochenen Statements und der betroffenen Daten ansehen; es gibt weitere „Fehler“ in den Daten.

Dump

u-id. DUMP.JOBLOG.S0C7#03.txt

Musterlösung (ohne Source-Code-Unterstützung)

wichtigste Informationen aus Dump:

```
<H1> I B M   F A U L T   A N A L Y Z E R   S Y N O P S I S

A system abend 0C7 occurred in module TES47 program TES47 at offset X'712'.

A program-interruption code 0007 (Data Exception) is associated with this abend
and indicates that:

  A decimal digit or sign was invalid.
The abend was caused by machine instruction FD94D150D160 (DP).

Recently referenced data items:
  The failing operand at 0001D4C8 is the result of a PACK instruction using the
  following zoned decimal data item which contains invalid data:

  Data Item . . . . . : BLL=0003+032
    At Address. . . . . : 3691108A
    Length. . . . . : X'9'
    Data Item Storage . . . : F0F0F0F0 F0F04C4C 4C  *000000<<<*

  Data Item . . . . . : BLL=0003+03C
    At Address. . . . . : 36911094
    Length. . . . . : X'9'
    Data Item Storage . . . : F0F0F0F0 F0F0F0F6 F6  *000000066*
```

suchen Abbruchstelle in Compileliste (von TES47)

find 0712 (wenn mit LIST umgewandelt)

➔ Befehl wird angezeigt

find hexloc (wenn mit NOLIST umgewandelt)

suchen Befehl mit Offset <= 0712

➔ `000102 0006FA COMPUTE 000103 000722 DISPLAY`

➔ also Zeile 102 beinhaltet den Befehl

➔ `000102 00101 COMPUTE FELD-ERGEBNIS-3 = FELD-5 / FELD-6`

An welcher Stelle sind die Felder im Dump zu suchen ...

```
000059         00058         10 FELD-5                  PIC  9(009).                CL**2 BLL=00003+032,0000032 9C
000061         00060         10 FELD-6                  PIC  9(009).                CL**2 BLL=00003+03C,000003C 9C
000037         00036         10 FELD-ERGEBNIS-3         PIC S9(009) BINARY.         CL*31 BLW=00000+044,0000044 4C
```

Suchen im Dump nach den Inhalten der Felder

```
Program and Entry Point Name: TES47
  At Address. . . . . . . . : 3691C7E8 (Module TES47 offset X'0')
  Program Length. . . . . . : X'BE4'
  Program Language. . . . . : COBOL (Compiled using IBM Enterprise COBOL for
                              z/OS and OS/390 V4 R1 M0 on 2012/05/24 at
                              17:11:27)

        Event 3 Program TES47 BLW=0000 (Address 369110F8)
        Event 3 Program TES47 GPR 4 (Address 369110F8)
69110F8                              E3C5E2F4 F7404040 *        TES47   *
6911100        +8  E3C5E2F6 F6404040 00000000 00000000 *TES66   ........*
6911110       +18  00000000 00000000 00000000 00000000 *................*
6911120       +28  00000000 00000000 00000000 F8F5F9F1 *............8591*
6911130       +38  F5F5F9F7 C4000000 00088C00 00000000 *5597D...........*
6911140       +48  00000000 00000000 00000000 00000000 *................*
        Lines 36911150-369111A0 (X'60' bytes) same as above
69111B0       +B8  00000000 00000000 00000000 0000     *..............  *...
```


cps4it

```
        Event 3 Program TES47 BLL=0003 (Address 36911058)
        Event 3 Program TES47 GPR 3 (Address 36911058)
6911058                          F0F0F0F0 F0F0F0F0 *        00000000*
6911060        +8  F14FC8C5 C9A1C5C5 C9D5E24F F0F0F0F0 *1|HEI~EEINS|0000*
6911070       +18  F0F0F0F2 F24FF0F0 F0F0F0F0 F0F0F24F *00022|000000002|*
6911080       +28  F0F0F0F0 F0F0F0F4 F44FF0F0 F0F0F0F0 *000000044|000000*
6911090       +38  4C4C4C4F F0F0F0F0 F0F0F0F6 F64FF0F0 *<<<|000000066|00*
69110A0       +48  40404040 40404040 00000000 00000000 *        ........*
69110B0       +58  00000000 00000000 00000000 00000000 *................*
69110C0       +68  00000000 00000000 00000000 00000000 *................*
69110D0       +78  00000000 00000000 36911000 000000F8 *........j.....8*
69110E0       +88  000000EC 00000000 00000000 00000000 *................*
69110F0       +98  00000000 00000000                   *........        *
```

Feld FELD-ERGEBNIS-3 beinhaltet also Low-Values (klar, weil es das Zielfeld ist); es ist binär und 4 Bytes lang.
Feld FELD-5 beinhaltet 000000<<<
Feld FELD-6 beinhaltet 000000066

Den Inhalt des Feldes FELD-5 hat Fault Analyzer schon als fehlerhaft angezeigt („wichtigste Informationen aus Dump“).

Der erste Teil der Aufgabe ist also gelöst.
Bei näherem Hinsehen auf die Dump-Inhalte fällt auf, dass da noch ein Inhalt **HEI~EEINS** vorhanden ist. Die Adresse, bei der dieser Inhalt anfängt, ist BLL Zelle 1 Offset 00A. Wenn wir uns an die erste Aufgabe erinnern, müsste da doch das Feld FELD-1 stehen. Dies verifizieren wir.

Extrakt aus Compile-Liste:

```
000051         00050         10 FELD-1                  PIC  9(009).                    CL**2 BLL=00003+00A,000000A 9C
```

Es stimmt also, das auf dieser Adresse FELD-1 zu finden ist. Schauen wir noch mal in den Programmcode:

```
   000098         00097      COMPUTE FELD-ERGEBNIS-1 = FELD-1 + FELD-2                    CL**5 33 51 53
   000099         00098      DISPLAY '* ERGEBNIS ADDITION       : ' FELD-ERGEBNIS-1       CL**5 33
   000100         00099      COMPUTE FELD-ERGEBNIS-2 = FELD-3 * FELD-4                    CL**5 35 55 57
   000101         00100      DISPLAY '* ERGEBNIS MULTIPLIKATION : ' FELD-ERGEBNIS-2       CL**5 35
   000102         00101      COMPUTE FELD-ERGEBNIS-3 = FELD-5 / FELD-6                    CL**5 37 59 61
```

```
000103         00102      DISPLAY '* ERGEBNIS DIVISION       : ' FELD-ERGEBNIS-3          CL**5 37
000104         00103      COMPUTE FELD-ERGEBNIS-4 = FELD-6 / FELD-7                       CL*31 39 61 63
```

Da erkennen wir, dass der Compute mit FELD-1 ***vor*** dem Compute mit FELD-6 steht. Beide Felder beinhalten ungültige nummerische Werte. Aber warum kommt es beim ersten Compute **nicht** zu einem S0C7?

Schauen wir uns den Inhalt von FELD-1 noch einmal an. Wir schreiben es in der Form, wie es im ISPF-Edit sichtbar ist:

```
CCCACCCDE
859155952
HEI.EEINS
```

H ist also C8, E ist C5 und so weiter.

Der COBOL-Compiler geht bei einer nummerischen Operation mit PIC n(9)- oder PIC Sn(9)-Feldern so vor, dass er Rechenfelder zunächst packt (in das Format PACKED DECIMAL / COMP-3 umformatiert) und dann die Rechenoperation ausführt. Danach macht er – wenn erforderlich – die Konvertierung in das Format des Zielfeldes. Auf unseren Fall angewendet heißt das, dass die Inhalte von FELD-1 und FELD-2 zu packen sind. Wie geht die Pack-Operation vor sich? Der Compiler nimmt jeweils das zweite Halbbyte und überträgt dies in sein temporäres Hilfsfeld. Er schreibt also in das Zielfeld die Halbbytes 859155952 und setzt in das letzte Halbbyte noch ein Vorzeichen, das C. Dieses Feld ist ***sauber*** nummerisch. Es kann also bei dieser Operation nicht zu einem S0C7 kommen. Diese Vorgehensweise wählt er, um einen zur Laufzeit möglichst schnellen Code zu erzeugen. Wären in irgendeinem der „unteren“ Halbbytes keine Ziffer, sondern A oder B,C,D,E, F, würde es zu einem S0C7 kommen. Wenn wir uns den Inhalt von FELD-5 ansehen, haben wir auch 3 Mal ein C im zweiten Halbbyte.

5 S0C4 – 1

Aufgabe

Suchen nach der Abbruchursache.

Dump

uid.DUMP.JOBLOG.S0C4#01.txt

Musterlösung (ohne Source-Code-Unterstützung)

wichtigste Informationen aus Dump:

```
- <H1> I B M   F A U L T   A N A L Y Z E R   S Y N O P S I S


A system abend 0C4 reason code X'4' occurred in module IGZCPAC at offset
X'716E'.

A program-interruption code 0004 (Protection Exception) is associated with this
abend and indicates that:

  An attempt was made to reference main storage that was not available in the
  configuration.

The abend occurred after executing machine instruction 0DEF (BRANCH AND SAVE) in
module TES47 program TES47 at offset X'7D2'.

NOTE: Source code information could not be presented because the search for a
      compiler listing or side-file was unsuccessful for program TES47.
```

Das ist sehr verwirrend! Was ist da los?

```
- <H1> I B M   F A U L T   A N A L Y Z E R   E V E N T   S U M M A R Y


The following events are presented in chronological order.

Event           Fail  Module   Program  EP
#  Type         Point Name     Name     Name     Event Location (*) Description
-- ------------ ----- -------- -------- -------- ------------------ ----------------------------------------
 1 Call               TES39    TES39    TES39    E+560              From SYS4.TEST.PGMLIB
 2 Call               IGZCPAC  n/a      IGZCFCC  E+2C0              INITIAL LOAD PHASE (COBPA C)); From LPA
 3 Call         ***** TES47    TES47    n/a      P+7D2              From SYS4.TEST.PGMLIB
 4 Execute            IGZCPAC  n/a      IGZCDSP  E+8A2              FORMAT AND PRINT OPERANDS OF DISPLAY VRB; From LPA
 5 Abend S0C4         IGZCPAC  n/a      IGZCDSP  E+8C6              FORMAT AND PRINT OPERANDS OF DISPLAY VRB; From LPA
```

Das heißt doch, dass der Fehler im Tes47 passiert sein muss. Aber da ist so viel kaputt, dass sogar der Status des Programms nicht mehr richtig angezeigt werden kann. Weiter:

Informationen aus SYSOUT:

```
CEE3204S The system detected a protection exception (System Completion Code=0C4).
         From compile unit TES39 at entry point TES39 at compile unit offset +00000560 at entry offset +00000560 at
         address 35B01180.
```

Das ist laut Eevnt Summary der Call im TES39. Also können wir fast sicher sein, dass im TES47 der Hund begraben liegt.

Also: Umwandlung mit der Compileoption SSRANGE und erneuter SUBMIT liefert, …

Ergebnis:

… dass es ein Tabellenüberlauf war. Man schaue sich die DISPLAYs an. ;-)

Hinweis: So sieht die Information im SYSOUT aus, wenn man mit SSRANGE umgewandelt hat:

```
  IGZ0006S The reference to table TAB-NR2 by verb number 01 on line 000122
           addressed an area outside the region of the table.
```

6 S0C4 – 2

Aufgabe

Suchen nach der Abbruchursache.

Dump

uid.DUMP.JOBLOG.S0C4#02.txt

Musterlösung (ohne Source-Code-Unterstützung)

wichtigste Informationen aus Dump:

```
- <H1> I B M   F A U L T   A N A L Y Z E R   S Y N O P S I S


A system abend 0C4 reason code X'4' occurred in module TES47 program TES47 at
offset X'968'.

A program-interruption code 0004 (Protection Exception) is associated with this
abend and indicates that:

  An attempt was made to reference main storage that was not available in the
  configuration.

The abend was caused by machine instruction 0E40 (MOVE LONG).

Recently referenced data items:

  Data Item . . . . . : BLL=0001+000
    At Address. . . . . : 70F0F0F0
    Length. . . . . : X'4'
    Data Item Storage . . . : n/a

NOTE: Source code information could not be presented because the search for a
      compiler listing or side-file was unsuccessful for program TES47.
```

cps4it

suchen Abbruchstelle in Compileliste (von TES47)

find 0968 (wenn mit LIST umgewandelt)
➔ Befehl wird angezeigt
find hexloc (wenn mit NOLIST umgewandelt)
suchen Befehl mit Offset <= 0968
➔ `000179 000954 MOVE` (also Zeile 179 beinhaltet den Befehl)
➔ `000179 00178 MOVE SPACES TO LINKAGE-ZUSATZ`

An welcher Stelle sind die Felder im Dump zu suchen ...

```
000065 00064 01 LINKAGE-ZUSATZ PIC X(5000). CL*32 BLL=00002+000 5000C
```

Das Feld ist definiert.

```
000068 00067 PROCEDURE DIVISION USING EINGABE-ZEILE LINKAGE-ZUSATZ.
```

Es ist in der USING Leiste.

Die Suche, ob mit dem Feld noch etwas getan wird liefert kein Ergebnis.

Also: Wir müssen also den CALL vom Aufrufer prüfen. Wer ruft auf?

wichtigste Informationen aus Dump (2):

```
- <H1> I B M   F A U L T   A N A L Y Z E R   E V E N T   S U M M A R Y


The following events are presented in chronological order.

Event            Fail  Module   Program  EP
#  Type          Point Name     Name     Name     Event Location (*) Description
-- ------------ ----- -------- -------- -------- ------------------ ------------------------------------------
 1 Call                TES39    TES39    TES39    E+560              From SYS4.TEST.PGMLIB
 2 Call                IGZCPAC  n/a      IGZCFCC  E+2C0              INITIAL LOAD PHASE (COBPA C)); From LPA
 3 Abend S0C4   ***** TES47    TES47    TES47    E+968              From SYS4.TEST.PGMLIB
```

Der Call ist auf Displacement 000560 zu suchen.

suchen CALL in Compileliste (von TES39)

find 0560 (wenn mit LIST umgewandelt)
➔ Befehl wird angezeigt
find hexloc (wenn mit NOLIST umgewandelt)
suchen Befehl mit Offset <= 0560

➔ `000077 000526 CALL` (also Zeile 77 beinhaltet den Befehl)
➔ `000077      1  00076           CALL TES47 USING EINGABE-ZEILE`

Ergebnis: Using-Leiste stimmt nicht überein.

7 S806

Aufgabe

Suchen nach der Abbruchursache.

Dump

uid.DUMP.JOBLOG.S806.txt

Musterlösung (ohne Source-Code-Unterstützung)

wichtigste Informationen aus Dump:

```
- <H1> I B M   F A U L T   A N A L Y Z E R   S Y N O P S I S


The system load of module TES66 failed. The load module was not found in the
standard MVS search path.

This is probably a user error.
```

Das sieht man auch im SYSOUT:

```
CEE3501S The module TES66    was not found.
          From compile unit TES47 at entry point TES47 at compile unit offset +00000932 at entry offset +00000932 at
          address 3601D0F2.
```

suchen Abbruchstelle in Compileliste (von TES47)

find 0932 (wenn mit LIST umgewandelt)
➔ Befehl wird angezeigt
find hexloc (wenn mit NOLIST umgewandelt)
suchen Befehl mit Offset <= 0932

Oder. Einfach nach „TES66“ in der Compileliste suchen. Weitere Analysen sollten nicht notwendig sein.

8 S0CB – 1

Aufgabe

Suchen nach der Abbruchursache.

Dump

uid.DUMP.JOBLOG.S0CB#01.txt

Musterlösung (ohne Source-Code-Unterstützung)

wichtigste Informationen aus Dump:

```
- <H1> I B M   F A U L T   A N A L Y Z E R   S Y N O P S I S

A system abend 0CB occurred in module TES47 program TES47 at offset X'744'.

A program-interruption code 000B (Decimal-Divide Exception) is associated with
this abend and indicates that:

  The divisor was zero in a signed decimal division.
The abend was caused by machine instruction FD94D150D160 (DP).

Recently referenced data items:
  The failing operand at 0001D4D8 is the result of the following numeric data
  item being zero:
  Data Item . . . . . . . . . : BLL=0003+046
    At Address. . . . . . . . : 36A1109E
    Length. . . . . . . . . . : X'9'
    Data Item Storage . . . . : F0F0F0F0 F0F0F0F0 F0  *000000000*
  Data Item . . . . . . . . . : BLL=0003+03C
    At Address. . . . . . . . : 36A11094
    Length. . . . . . . . . . : X'9'
    Data Item Storage . . . . : F0F0F0F0 F0F0F0F6 F6  *000000066*

NOTE: Source code information could not be presented because the search for a
      compiler listing or side-file was unsuccessful for program TES47.
```

suchen Abbruchstelle in Compileliste (von TES47)

find 0744 (wenn mit LIST umgewandelt)

➔ Befehl wird angezeigt

find hexloc (wenn mit NOLIST umgewandelt)

suchen Befehl mit Offset <= 0744

➔ `000104 00072C COMPUTE` (also Zeile 104 beinhaltet den Befehl)

➔ `000104  00103  COMPUTE FELD-ERGEBNIS-4 = FELD-6 / FELD-7`

An welcher Stelle sind die Felder im Dump zu suchen ...

```
000039   00038   10 FELD-ERGEBNIS-4   PIC S9(009) BINARY.   CL*31 BLW=00000+049,0000049 4C
000061   00060   10 FELD-6            PIC  9(009).          CL**2 BLL=00001+03C,000003C 9C
000063   00062   10 FELD-7            PIC  9(009).          CL*31 BLL=00001+046,0000046 9C
```

Suchen im Dump nach den Inhalten der Felder

```
         Event 3 Program TES47 BLW=0000 (Address 36A110F8)
         Event 3 Program TES47 GPR 4 (Address 36A110F8)
36A110F8                          E3C5E2F4 F7404040 *        TES47   *
. . .
36A11140       +48  00000000 00000000 00000000 00000000 *................*
```

Die anderen Feldinhalte wurden ja schon angezeigt, aber suchen wir sie trotzdem im Dump:

```
         Event 3 Program TES47 BLL=0003 (Address 36A11058)
         Event 3 Program TES47 GPR 3 (Address 36A11058)
36A11058                          F0F0F0F0 F0F0F0F0 *        00000000*
36A11060        +8  F14FF0F0 F0F0F0F0 F0F0F14F F0F0F0F0 *1!000000001!0000*
36A11070       +18  F0F0F0F2 F24FF0F0 F0F0F0F0 F0F0F24F *00022!000000002!*
36A11080       +28  F0F0F0F0 F0F0F0F4 F44FF0F0 F0F0F0F0 *000000044!000000*
36A11090       +38  F0F0F54F F0F0F0F0 F0F0F0F6 F64FF0F0 *005!000000066!00*
36A110A0       +48  F0F0F0F0 F0F0F0F1 00000000 00000000 *00000001........*
```


cps4it

Nun muss danach geforscht werden, wer diesen Inhalt liefert.

Erster Hinweis:
FELD-7 steht in der Linkage. Bei der weiteren Analyse des Programms TES39 findet man, dass das Feld FELD-7 direkt von den gelesenen Eingabedaten stammt.

9 S0CB – 2

Aufgabe

Suchen nach der Abbruchursache.

Dump

uid.DUMP.JOBLOG.S0CB#02.txt

Musterlösung (ohne Source-Code-Unterstützung)

wichtigste Informationen aus Dump:

```
- <H1> I B M   F A U L T   A N A L Y Z E R   S Y N O P S I S


A system abend 0CB occurred in module TES39 program TES39 at offset X'512'.

. . .

Recently referenced data items:

  The failing operand at 0001D130 is the result of the following numeric data
  item being zero:

  Data Item . . . . . . . . : BLW=0000+02C
    At Address. . . . . . . : 36A1106C
    Length. . . . . . . . . : X'9'
    Data Item Storage . . . : F0F0F0F0 F0F0F0F0 F0  *000000000*

  Data Item . . . . . . . . : BLW=0000+022
    At Address. . . . . . . : 36A11062
    Length. . . . . . . . . : X'9'
    Data Item Storage . . . : F0F0F0F0 F0F0F0F0 F1  *000000001*

NOTE: Source code information could not be presented because the search for a
      compiler listing or side-file was unsuccessful for program TES39.
```

suchen Abbruchstelle in Compileliste (von TES39)

find 0512 (wenn mit LIST umgewandelt)

➔ Befehl wird angezeigt

find hexloc (wenn mit NOLIST umgewandelt)

suchen Befehl mit Offset <= 0512

➔ `000075 0004FE COMPUTE` (also Zeile 75 beinhaltet den Befehl)

➔ `000075 1 00074 COMPUTE FELD-3 = FELD-1 / FELD-2 CL*74 33 29 31`

Aber: Schauen wir den Code mal genauer an – TES39:

```
000069          00068      WHEN I1-MAX-EINGABE = 6                                  CL*74 25
000070          00069 *          FELD-1 NACH FELD-2 UEBERTRAGEN UND ZURUECK GEBEN LASSEN  CL*74
000071       1  00070            MOVE 1     TO FELD-1                             CL*74 29
000072       1  00071            MOVE 0     TO FELD-2                             CL*74 31
000073       1  00072            CALL TES47 USING BY CONTENT EINGABE-ZEILE         CL*74 23 24
000074       1  00073            END-CALL                                         CL*74
000075       1  00074            COMPUTE FELD-3 = FELD-1 / FELD-2                 CL*74 33 29 31
```

Und in TES47:

```
000077          00076      EVALUATE (I1-MAX-EINGABE)                              CL**6 49
…
000083       1  00082      WHEN     (6) PERFORM FKT-6                             CL*34 187
…
000187          00186  FKT-6 SECTION.                                             CL*34
000188          00187 *                                                           CL*34
000189          00188      MOVE FELD-1   TO  FELD-2                               CL*34 51 53
```

Da müsste doch in FELD-2 der Wert von FELD-1 stehen. Das war die Absicht des Programmierers.

Er hat aber den CALL falsch geschrieben:

```
CALL TES47 USING BY CONTENT EINGABE-ZEILE
```

10 S0CB – 3

Aufgabe

Suchen nach der Abbruchursache.

Dump

uid.DUMP.JOBLOG.S0CB#03.txt

Anmerkungen

Zunächst ein paar Anmerkungen zu diesem Fehler. Es taucht in vielen Programmen eine Logik auf der Art, dass bei einer komplexen fachlichen Überprüfung versucht wird, eine komplette Prüfung vorzunehmen. Beispiel Compiler: Wenn ein Compiler nach jedem Fehler sofort seine Arbeit beenden würde, wären wir fast nur mit dem Warten auf die Umwandlungen beschäftigt. Daher hat es sich eingebürgert, dass Schalter für bestimmte Fehler(-Typen) gesetzt werden. Diese werden am Anfang initialisiert und am Ende der Durchführung werden alle Schalter geprüft. Die implementierte Logik in dem benutzten Programm basiert auf einer solchen Logik. Dabei ist bei der Programmierung aber ein kleiner Fehler passiert. Und diesen gilt es zu finden.

Erläuterungen zum Programm TES46

Das Programm besteht aus 3 Teilen, von denen nur 2 für uns relevant sind.

1. Teil 1 besteht aus der Section ANALYSE-SETZEN. Diese Section initialisiert alle unsere Fehler-Schalter mit dem Wert 1.
2. Teil 2 besteht aus der Funktion, bei der Fehler auftreten könnten und dann die Schalter verändert werden. In unserem Fall werden in dieser Section keine Fehler gefunden. Die Schalter sollten also alle auf 1 stehen. Diese Section braucht uns also nicht zu interessieren.
3. Teil 3 besteht aus der Section ANALYSE-PRUEFEN. Dort werden die Schalter abgefragt. Sollte irgendein Schalter nicht auf dem Wert 1 stehen, wird ein Fehlerzähler um den Wert 1 hochgezählt. Am Ende aller Prüfungen wird dieser Fehlerzähler geprüft. Steht dieser nicht auf 0, wird ein harter Abbruch (Division durch 0) erzwungen.

Es gilt also heraus zu finden, warum – obwohl angeblich richtig kodiert wurde – eine Division durch 0 passiert.

Musterlösung (ohne Source-Code-Unterstützung)

wichtigste Informationen aus Dump:

```
- <H1> I B M   F A U L T   A N A L Y Z E R   S Y N O P S I S


A system abend 0CB occurred in module TES46 program TES46 at offset X'E2E'.

A program-interruption code 000B (Decimal-Divide Exception) is associated with
this abend and indicates that:

  The divisor was zero in a signed decimal division.

The abend was caused by machine instruction FD94A148A140 (DP).

Recently referenced data items:

  The failing operand at 0001D4B8 is the result of the following numeric data
  item being zero:

  Data Item . . . . . . . . : BLW=0002+00A
    At Address. . . . . . . : 36A1317A
    Length. . . . . . . . . : X'9'
    Data Item Storage . . . : 40404040 40404040 40  *         *

  Data Item . . . . . . . . : BLW=0002+17A
    At Address. . . . . . . : 36A132EA
    Length. . . . . . . . . : X'5'
    Data Item Storage . . . : 00000000 1C  *.....*

NOTE: Source code information could not be presented because the search for a
      compiler listing or side-file was unsuccessful for program TES46.
```

Okay. Nach der Vorgeschichte ist das ja klar. Aber wo ist der krumme Hund? Der Referent behauptet ja, er habe richtig kodiert und die Division durch 0 dürfe nicht auftreten.

Also: Die Fehler heißen alle ZAHL-G, -H, -I, -J, -K usw. Diese werden abgefragt. In der Section Analyse-setzen heißen sie auch so. Hmhm. Man könnte ja auch ein wenig schöner kodieren, dass die Sachen schön untereinander stehen. Hmhm. Wenn man jetzt mit View in der Compileliste ist (nicht mit Browse) kann man das ja mal machen. Siehst Du schon was? Okay. Weiter.

Schauen wir mal in die Cross-Referenz:

```
 98   ZAHL-A
 99   ZAHL-B
102   ZAHL-C
103   ZAHL-D
104   ZAHL-E
105   ZAHL-F
108   ZAHL-G . . . . . .  M194 208
109   ZAHL-H . . . . . .  M195 214
110   ZAHL-I . . . . . .  220
111   ZAHL-J . . . . . .  M197 226
112   ZAHL-K . . . . . .  M198 232
113   ZAHL-L . . . . . .  M199 238
114   ZAHL-M . . . . . .  M200 244
115   ZAHL-N . . . . . .  M201 250
100   ZAHL-1 . . . . . .  M196
101   ZAHL-2
```

Noch Fragen?

Fazit 1: Es lohnt sich, den Code sauber auszurichten.
Fazit 2: Es lohnt sich, Felder mit gleicher Funktion gleich zu benennen und erst „hinten“ zu unterscheiden.
Fazit 3: Es lohnt sich, zu überlegen, wo welche Information stehen könnte, die mir bei der Suche behilflich sein könnte.
Fazit 4: Präge Dir ein, welche Informationen in der Compileliste vorhanden sind.
Fazit 5: Glaube niemals dem Referenten, ehe Du seine Aussagen nicht überprüft hast. ;-)