Indexes

| **Number** | **Abbreviation** | **Description** |
| --- | --- | --- |
| 1 | FTCC | First TC crashes |
| 2 | NSEF | No support for external C library functions (except for C file handling functions) |
| 3 | NSFF | No support for external file-handling functions |
| 4 | NSFP | No support for function pointers |
| 5 | NSGP | No support for global void pointer |
| 6 | NSLS | No support for local static variable |
| 7 | NSSF | No support for symbolic file |
| 8 | NSSP | No support for symbolic pointers |
| 9 | RT1 | Timeout1 is reached (e.g., due to the infinite loop caused by unrealistic symbolic input values and/or too many symbolic variables) |
| 10 | TIWC | Target program’s Illegal write to the CROWN’s data structure that causes failure |
| 11 | UB | Existence of unreachable branches |
| 12 | USV | Uncovered branch caused by the unrealistic symbolic input values |
| 13 | WDFS | Weakness of dfs |

| **Reason** | **ID** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| FTCC | [Flex-1](#i4soqre4e40m) | [Flex-6](#z81bljdi5kw3) | [Flex-7](#rh1splz5ef2a) | [Flex-8](#ev1pu5mo4zfy) | [Flex-9](#rjjngcwb21ap) | [Flex-10](#1hjmrpkbn7aw) | [Flex-11](#omckr5hfgacm) |
| [Flex-12](#jx8xq5dw4fwb) | [Flex-18](#2gszpq5b5f4d) | [Flex-19](#4a9yvsnhzyv3) | [Flex-20](#kme85nxhft71) | [Flex-22](#yk65yzuegfhf) | [Flex-23](#41haaftjrvvl) |  |
| NSSF | [Flex-2](#vuma2abl8yf7) | [Flex-3](#qhdo8ixsf3mn) | [Flex-4](#9cxp0sg7wwq6) | [Flex-5](#ql130wuc2qyu) | [Flex-8](#ev1pu5mo4zfy) | [Flex-9](#rjjngcwb21ap) | [Flex-10](#1hjmrpkbn7aw) |
| [Flex-11](#omckr5hfgacm) | [Flex-12](#jx8xq5dw4fwb) | [Flex-18](#2gszpq5b5f4d) | [Flex-19](#4a9yvsnhzyv3) | [Flex-20](#kme85nxhft71) | [Flex-22](#yk65yzuegfhf) | [Flex-23](#41haaftjrvvl) |
| [Flex-24](#lsroq8read4g) |  |  |  |  |  |  |
| NSSP | [Flex-15](#8b7wb6pavthq) | [Flex-16](#lm82vvg7pjhs) | [Flex-17](#8tlgoo5qxgt1) | [Flex-25](#by946e42msza) | [Flex-28](#thzq71hd9to0) | [Flex-30](#mmsi1kk0lrtw) |  |
| RT1 | [Flex-26](#80itv6wpcqzm) | [Flex-27](#y4fnb8aryhn4) | [Flex-29](#v0iqsontjzvm) |  |  |  |  |
| USV | [Flex-13](#sm4tvxypyyl7) | [Flex-14](#i0n1el715mmg) | [Flex-21](#do7rky5snxji) |  |  |  |  |

Flex-1

Function Name: main

Reason: FTCC

Line No: 200-296 (crash at line 213)

Description: No more TCs can be generated as the execution of first TC crashes

1. Line 213: argv is a global pointer (char \*\*) and CROWN allocates 1 element for this pointer (i.e., argv = malloc(1\*sizeof(char\*)))
2. Line 213: this line tries to access argv[2], where an out of bound memory access happens
   1. No more TCs can be generated.

200. int main(int argc, char \*\*argv){

...

213. **if**(**argv[2]**[0] != '-' &&...)

...

296. }

Flex-2

Function Name: flexend

Reason: NSSF

Line No: 313-321

Description:

1. Line 312: skelfile is a pointer to the structure FILE, and CROWN assigns NULL to this pointer. So the “then” branch cannot be explored

312. if(skelfile !=NULL) // skelfile = NULL

313. {

**… // not covered**

321. }

Flex-3

Function Name: flexend

Reason: NSSF

Line No: 324-333

Description:

1. Line 323: CROWN tries to satisfy the condition of “if” statement and explore the “then” branch
2. Line 326: the execution crashes when calling function flexfatal, which tries to write string into **NULL** (FILE \*) variable err
   1. The execution of “then” branch (line 324-333) cannot be recorded by gcov

323. if(cond) // cond is the condition of “if” statement

**324. { // not covered begin**

**...**

**326. flexfatal(...); // == fprintf(err,...) where err=NULL**

**333. } // not covered end**

Flex-4

Function Name: flexend

Reason: NSSF

Line No: 335-352

Description: Refer to [Flex-2](#vuma2abl8yf7)

Flex-5

Function Name: flexend

Reason: NSSF

Line No: 355-506

Description: Refer to [Flex-3](#qhdo8ixsf3mn) (crash at line 356)

Flex-6

Function Name: flexinit

Reason: FTCC

Line No: 518-906 (crash at 560)

Description: Refer to [Flex-1](#i4soqre4e40m)

Flex-7

Function Name: readin

Reason: FTCC

Line No: 911-983 (crash at line 917)

Description:

1. Line 917: Crash (dereference a NULL pointer) happens when executing this line.
   1. char \* s1 = "INITIAL", char \* s2 = NULL;

**917. while(\*s1 && \*s2)**

**...**

Flex-8

Function Name: ccladd

Reason: FTCC, NSSF

Line No: 1126-1157 (crash at line 1132)

Description:

1. Line 1132: err is a FILE pointer and CROWN assigns NULL to this pointer.
   1. The first TC executes this line and crash happens as CROWN tries to write string into the **NULL** FILE (dereference NULL pointer)

**1132. fprintf(err, ...);**

Flex-9

Function Name: list\_character\_set

Reason: FTCC, NSSF

Line No: 1210-1240 (crash at line 1216)

Description:

1. Line 1216: file is a global FILE pointer, and CROWN assigns NULL to this pointer
   1. crash happens when executing function putc (dereference the NULL pointer file)

**1216. putc( '[', file ); // dereference NULL pointer (file)**

Flex-10

Function Name: dump\_associated\_rules

Reason: FTCC, NSSF

Line No: 1385-1423 (crash at line 1412)

Description:

1. Line 1412: file is a global FILE pointer, and CROWN assigns NULL to this pointer
   1. Crash happens when executing fprintf (writing string into **NULL** file)

**1412. fprintf( file, " associated rule line numbers:" ); // crash**

Flex-11

Function Name: epsclosure

Reason: FTCC, NSSF

Line No: 1486-1602 (crash at 1544)

Description: Refer to [Flex-8](#ev1pu5mo4zfy)

Flex-12

Function Name: ntod

Reason: FTCC, NSSF

Line No: 1633-2005 (crash at 1655)

Description:

1. Line 1655: err is a FILE pointer and CROWN assigns NULL to this pointer.
   1. The first TC executes this line and crash happens as CROWN tries to write string into the **NULL** FILE (dereference NULL pointer)

**1655: fprintf(err, ...);**

Flex-13

Function Name: snstods

Reason: USV

Line No: 2128-2129

Description:

1. Line 2127: nacc is a symbolic integer, to explore the “then” branch of “for” statement, CROWN outputs 1073741824 and assigns it to nacc
2. Line 2128: accset is a pointer to integer, and CROWN assigns only **1** element for it (accset = malloc(1\*sizeof(int)))
   1. crash happens as this line can cause out of bound memory access (e.g., try to access accset[10000])
   2. the coverage of line 2128-2129 cannot be recorded by gcov

2127. for(int i=1; i<=nacc; ++i){ // begin of for loop

**2128. if(accset[i]< j){ j = accset[i];}**

2129. } // end of for loop

Flex-14

Function Name: symfollowset

Reason: USV

Line No: 2158-2224

Description: refer to [Flex-13](#sm4tvxypyyl7)

Flex-15

Function Name: genctbl

Reason: NSSP

Line No: 2730-2738

Description:

1. Line 2720: base is a symbolic variable (assigned with **0**), chk is a pointer and CROWN assigns one element for it
   1. At line 2720, chk[0] is **always assigned with -1** as base is not used by other branch conditions and CROWN does not support symbolic array index (i.e., the value of base cannot be altered).
   2. Line 2727: the “else” branch of “if” statement cannot be covered.

… // base is not used by other branch conditions

2720. chk[base] = -1;

… // chk[0] is not altered

2727. if(chk[0] == -1) {...}

**else {...} // not covered**

Flex-16

Function Name: gen\_next\_state

Reason: NSSP

Line No: 3128-3136

Description:

1. Line 3127: nultrans is a global pointer and CROWN allocates 1 element for it (i.e., nultrans = malloc(1\*sizeof(int)))
   1. The “then” branch of line 3127 (i.e., 3128-3136) cannot be covered as CROWN cannot alter the nultrans to NULL

3127. if(!nultrans) { // nultrans is not NULL

**… // not covered**

3136.}

Flex-17

Function Name: gen\_NUL\_trans

Reason: NSSP

Line No: 3205-3241

Description: Refer to [Flex-16](#lm82vvg7pjhs)

Flex-18

Function Name: gentabs

Reason: FTCC, NSSF

Line No: 3293-3536 (crash at line 3304)

Description:

1. Line 3304: err is a FILE pointer and CROWN assigns NULL to it. fprintf tries to dereference err, which causes a crash. And the first TC executes this line.
   1. No more test cases can be generated

**3304: fprintf(err, ...);**

Flex-19

Function Name: make\_tables

Reason: FTCC, NSSF

Line No: 3567-3989 (crash at line 3645)

Description: Refer to [Flex-18](#2gszpq5b5f4d)

Flex-20

Function Name: add\_action

Reason: FTCC, NSSF

Line No: 4029-4044 (crash at line 4038)

Description: Refer to [Flex-18](#2gszpq5b5f4d)

Flex-21

Function Name: cshell

Reason: USV

Line No: 4240-4246

Description:

1. Line 4239: CROWN tries to explore the “then” branch of “if statement”
2. Line 4241: v is an array and CROWN only allocates 1 element for this array
3. Line 4241: jg is a symbolic integer, and CROWN assigns 16 to it
   1. an out-of-bound memory access happens as 16 > (length(v) == 1), the execution of “then” branch of line 4239 cannot be recorded by gcov

… // jg is used by other branch conditions

4239. if(cond) // cond is the condition of “if” statement

**4240. { // not covered begin**

**... if(v[jg] == 0){...} // crash (len(v) = 1, and jg = 16)**

**4246. } // not covered end**

Flex-22

Function Name: dumpmachine

Reason: FTCC, NSSF

Line No: 4908-4943 (crash at line 4933)

Description: Refer to [Flex-18](#2gszpq5b5f4d)

Flex-23

Function Name: finish\_rule

Reason: FTCC, NSSF

Line No: 4958-5026 (crash at line 4977)

Description: Refer to [Flex-18](#2gszpq5b5f4d)

Flex-24

Function Name: mkor

Reason: NSSF

Line No: 5218-5255

Description:

1. Line 5217: CROWN outputs 1 TC to satisfy the “if” statement.
   1. Line 5249: the current TC executes this line, where crash happens as err is FILE pointer (assigned with NULL) and this line tries to dereference err

5217. if(cond) // cond is the condition of “if” statement

**5218. { // not covered begin**

**... // err is not altered**

**5249. fprintf(err, ...);**

**...**

**5255. } // not covered end**

Flex-25

Function Name: mkentry

Reason: NSSP

Line No: 7343-7360

Description:

1. Line 7342: variable state is a symbolic array and CROWN only declares its first element as symbolic (i.e., SYM\_int(state[0]))
2. Line 7342: variable i is greater than 1, so state[i] is not a symbolic variable and its value is -1
   1. The “else” branch cannot be covered

… // i is greater than 1 (ref: 7291)

7342. if(state[i] !=-1) {...}

**7343-7360. else {...} // not covered**

Flex-26

Function Name: yylex

Reason: RT1

Line No: 7678-7839 (infinite loop happens at line 7686)

Description: The execution of the first TC has an infinite loop, which reaches timeout1 so that no more TCs can be generated.

1. Line 7686: index is a symbolic integer (assigned with 0)
   1. “while” loop is an infinite loop because a[1] != 0 and b[0] = 0 are always satisfied

**40. #include<ddd> // const** int a[1659] = {0, 1,...}; // a[1] = 1

**41. #include<eee> // const** int b[470] = {0, 407,...}; // b[0] = 0

**… // index has not been changed at all.**

7686. toktype = flexscan(); // defined in L10598

// flexscan contains the following statements and

// while(a[index + 1] != index){ index = b[index];} }

Flex-27

Function Name: flexscan

Reason: RT1

Line No: 10573-11372 (infinite loop happens at line 10598)

Description: Refer to [Flex-26](#80itv6wpcqzm)

Flex-28

Function Name: yy\_get\_previous\_state

Reason: NSSP

Line No: 10497-10511

Description:

1. Line 10496: variable a and b are symbolic pointer, and the value of a is greater than b
   1. CROWN cannot alter a or b to satisfy the condition (a < b) as the value of a and b are not symbolic
   2. So the “then” branch of “for” statement cannot be covered

11496. for(; a < b;a++) // a = char\*, b = char\*

**11497-10511. {...} // not covered**

Flex-29

Function Name: yy\_try\_NUL\_trans

Reason: RT1

Line No: 11524-11549 (infinite loop happens at line 11539)

Description: Refer to [Flex-26](#80itv6wpcqzm)

Flex-30

Function Name: input

Reason: NSSP

Line No: 11620-11657

Description: Line 11614: Refer to [Flex-28](#thzq71hd9to0)

**Reference**

1. 12 high-level reasons that are discovered before: