Reasons & Index

| **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 unrealistic symbolic input values / 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 in unit-testing |
| 12 | USV | Uncovered branch caused by the unrealistic symbolic input values |
| 13 | WDFS | Weakness of dfs |

| **Reason** | **ID** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| FTCC | [Gzip-3](#u4123sdyw5sn) | [Gzip-20](#rdmv93v9h2pw) |  |  |  |  |  |
| NSEF | [Gzip-1](#t0hgl7az9pw2) | [Gzip-2](#up3d7p3mgguu) | [Gzip-4](#2hprz1lzonc6) | [Gzip-12](#kix.lhvttifdsqmd) | [Gzip-13](#5v5356tyjfz5) | [Gzip-16](#m68u5qxvh64l) | [Gzip-18](#fk91tg4kdrd) |
| [Gzip-19](#phomy9y4slnv) | [Gzip-21](#4whay8rmrm3) | [Gzip-22](#sexgcl1ik3ox) | [Gzip-23](#fgpsr9a0ugwl) | [Gzip-24](#loglaw1kl34h) | [Gzip-25](#44rz8sownp5o) | [Gzip-26](#slxbjd74mqph) |
| [Gzip-27](#w8tgrngymzlv) | [Gzip-28](#gb36nazg9z7o) | [Gzip-29](#v642hr793dev) | [Gzip-30](#47ysphpb5p2w) | [Gzip-31](#yb47x73hz1gd) | [Gzip-32](#dj344n0szev) | [Gzip-33](#ex65afyu7201) |
| [Gzip-34](#kix.mqtj6jb4ge1b) | [Gzip-35](#108jwts5zhsj) | [Gzip-36](#sniicu1av6wc) | [Gzip-37](#kix.y4bzfrr9uwtd) | [Gzip-38](#kix.bn5a8akpgsyl) | [Gzip-39](#kix.2ru3elh2ozqj) | [Gzip-40](#kix.z6anlpsu2zf3) |
| [Gzip-41](#kix.yrba8w5nhkob) | [Gzip-42](#kix.i1en94ifpoc3) | [Gzip-43](#kix.cflnp3wvyhtj) | [Gzip-44](#kix.ovz2ptn7xe7c) | [Gzip-45](#kix.wqp5ibp5hlxz) |  |  |
| TIWC | [Gzip-3](#u4123sdyw5sn) | [Gzip-6](#l870u27pz4c6) | [Gzip-10](#5phdc1nljar9) | [Gzip-11](#9v9eropaqenf) |  |  |  |
| USV | [Gzip-7](#tpzafib3suy7) | [Gzip-8](#yd4wiofbxjt) | [Gzip-9](#7yqjqx5h5cd8) | [Gzip-14](#9mqlamqmdgm3) | [Gzip-15](#ycnx17v91u6s) | [Gzip-17](#h629mjj423k3) |  |

Gzip-1

Function Name:lm\_init

Reason: NSEF

Line No: 543-550

Description:

1. Line 537: read\_buf calls an external function and assigns concrete 0 to a
   1. Line 543-550: The “else” branch of “if” statement cannot be covered as the value of a is 0

537. unsigned int a = read\_buf(...);

538.

539. if(a == 0) {...}

**543-550. else {...} // not covered**

Gzip-2

Function Name:fill\_window

Reason: NSEF

Line No: 773

Description: Refer to Gzip-2

Gzip-3

Function Name: deflate

Reason: FTCC, TIWC

Line No: 868-951

Description: The execution of first TC crashes and CROWN cannot generate more than 1 TCs

1. Error message:

Error: Argument #x00000003 at position 1 does not match declaration (declare-fun bvashr ((\_ BitVec 64) (\_ BitVec 64)) (\_ BitVec 64))

Gzip-4

Function Name: huft\_build

Reason: NSEF

Line No: 1357-1362

Description:

1. Line 1356: Function malloc mostly returns non-NULL value, which makes the “then” branch unexecuted

1356. **if**(malloc(**sizeof**(**struct** huft)) == NULL)

**1357-1362.{ ... } // not covered**

Gzip-5

Function Name: inflate\_codes

Reason: USV

Line No: 1478-1485

Description:

1. Line 1483: CROWN assigns 4296947295 to symbolic unsigned integer a, which makes the execution reach timeout1
   1. The execution of the “then” branch of line 1477 cannot be recorded by gcov

**if**(a != 0){ **// condition of if statement, not covered begin**

… // a is not altered

**1483. flush\_window();**

**/\***

**\* void flush\_window(){**

**\* ...**

**\***

**\* while(a--) {...} // timeout1 is reached**

**\* ...**

**\* }**

**\*/**

**1485. } // not covered end**

Gzip-6

Function Name: inflate\_dynamic

Reason: TIWC

Line No: 1709-1814

Description:

1. CROWN TC generator was terminated due to the memory confilct, and the following is the error message:

Error: Argument x65 at position 1 does not match declaration (declare-fun bvshl ((\_ BitVec 64) (\_ BitVec 64)) (\_ BitVec 64))

Gzip-7

Function Name:pqdownheap

Reason: USV

Line No: 2384

Description:

1. Line 2381: k is a symbolic variable, and local variable j is assigned by a formula (k << 1)
2. Line 2384: to explore the “then” branch of “if” statement (i.e., k<<1 < heap\_len), CROWN outputs unrealistic values for both k and heap\_len
   1. k = 1342177279, heap\_len = 1610612736
   2. The execution for current TC crashes at line 2380, as heap only contains 573 elements (1342177279 >> 573)
   3. The “then” branch of line 2384 cannot be covered.

2380. int v = heap[k]; // heap is an global array whose length is 573

2381. int j = k << 1;

… // j is not altered

2384. if(j < heap\_len)

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

Gzip-8

Function Name:build\_tree

Reason: USV

Line No: 2551-2557

Description:

1. Line 2550: To cover the “then” branch of “for” statement, the value of symbolic variable elems should be greater than 1
2. Line 2551: Crash happens as this line will try to access the second element of tree, which has 1 element only (tree = malloc(**1**\*sizeof(\*tree)))
3. The coverage information of “then” branch (line 2551-2557) of “for” statement cannot be recorded by gcov

2550. for(int n=0; n < elems; n++){

**2551. if(tree[n].Freq != 0){...} // not covered begin**

**…**

**2557. } // not covered end**

Gzip-9

Function Name: compress\_block

Reason: USV

Line No: 2949-2977

Description: CR2 outputs an unrealistic variable that makes execution crash.

1. Line 2949: To cover the “then” branch of “if” statement, CR2 assigns 4296947295 to symbolic variable last\_lit, which makes crash happen at line 2951.
   1. Because inbuf only contains 32832 elements
2. The coverage information between 2949 and 2977 cannot be recorded.

2942. unsigned lx = 0; // lx is a local variable

… // lx is not altered

2949. **if**(last\_lit != 0)

**do**{

**...**

**// last\_lit = 4296947295**

**2951. lc = inbuf[lx++]; // inbuf = uchar[32832] , crashes**

**...**

2977. }**while**( lx < last\_lit);

Gzip-10

Function Name: unlzw

Reason: TIWC

Line No: 3210-3211

Description:

1. Error message:

Error: Argument #x00000003 at position 1 does not match declaration (declare-fun bvashr ((\_ BitVec 64) (\_ BitVec 64)) (\_ BitVec 64))

Gzip-11

Function Name: unlzw

Reason: TIWC

Line No: 3262-3354

Description:

1. refer to Gzip-10

Gzip-12

Function Name:read\_tree

Reason: NSEF

Line No: 3461-3462, 3467-3469, 3487-3493

Description:

1. Line 3459: get\_byte assigns **concrete** 0 to variable max\_len
2. Line 3460: the “then” branch (line 3461-3462) cannot be covered as max\_len < 25
3. Line 3466: the “then” branch (line 3467-3469) cannot be covered as max\_len < 1
4. Line 3486: the “then” branch (line 3487-3493) cannot be covered as max\_len < 1

3459. int max\_len = get\_byte(); // max\_len = 0

3460. if(max\_len > 25)

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

… // max\_len is not altered

3466. for(int len=1; len < max\_len; len++)

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

… // max\_len is not altered

3486. for(int len=1; len < max\_len; len++)

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

Gzip-13

Function Name: unpack

Reason: NSEF

Line No: 3552-3590

Description:Refer to Gzip-12

Gzip-14

Function Name:check\_zipfile

Reason: USV

Line No: 3657-3675

Description:

1. Line 3654: To cover the “else” branch, line 3567 must be executed, where crash happens.
   1. Because h is a global pointer and CROWN only allocates **1** element for it, but line 3657 tries to get the **9th** element of h (out of bound memory access)
   2. The coverage of line 3567-3675 cannot be recorded.

3654. if (cond) // cond is the condition of if statement

3655-3656 {... exit(1);} // then branch is covered and exits normally

**3657. method = h[8]; // crash**

**3658-3675. … // not covered**

Gzip-15

Function Name: unzip

Reason: USV

Line No: 3706-3712

Description:

1. Line 3705: Function inflate is called, where symbolic variables inptr and insize are used (initial value are 0)
   1. The “else” branch is covered (inptr = insize = 0), and the execution terminates by calling function exit
   2. To cover the “then” branch, CROWN outputs unrealistic values
      1. inptr = 1229979648, insize = 1229979649
      2. Execution crashes as array inbuf only contains 32832 (<<1229979648) elements
2. Line 3706-3712 cannot be covered.

3705. int res = inflate(); // inbuf.length = 32832, inptr = 1229979648

**/\***

**\* int inflate(){**

**\* … // inptr and insize are not altered**

**\* if(inptr < insize){return inbuf[inptr];}**

**\* else {exit(1);}**

**\* ...**

**\* }**

**\*/**

**3706-3712. ... // not covered**

Gzip-16

Function Name: fill\_inbuf

Reason: NSEF

Line No: 3852-3853

Description:

1. Line 3849: An external function open is called and returns -1, which is assigned to variable len
2. Line 3850: Only the “then” branch can be covered.

3849. int len = read(...); // return -1

3850. if(len == -1) {...}

**3852-3853. else{...} // not covered**

Gzip-17

Function Name:flush\_window

Reason: USV

Line No: 3882-3890

Description:

1. Line 3882: To execute this line, CROWN assigns symbolic unsigned integer variable a with 4296947295 (to satisfy the condition “!a==0”)
2. Line 3882: The “while” loop of function updcrc is executed until timeout1 is reached (because the value of a is 4296947295)
   1. Line 3882-3890 cannot be covered.

3881. if(a == 0) return;

3882. ulg updcrc(window, a);

// updcrc(uch\* w, int a){

// … // a is not altered

// while(a--){...}

// }

…

**3883-3890. … // not covered**

Gzip-18

Function Name:base\_name

Reason: NSEF

Line No: 3932

Description:

1. Line 3932: strrchr is an external function and returns NULL, which makes CROWN cover the “then” branch only.

3932. **if((**p = strrchr(...)) != NULL) {...}

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

Gzip-19

Function Name: add\_envopt

Reason: NSEF

Line No: 4018-4057

Description:

1. Line 4016: getenv returns NULL, which makes CR2 explore the “then” branch of 4017 and execute the “return” statement.
2. Line 4018-4057 cannot be covered

// env is a symbolic pointer

4016. env = getenv(env);

4017. **if**(env==NULL) **return** NULL;

4018-4057.  **//not covered**

Gzip-20

Function Name: zip

Reason: FTCC

Line No: 4214-4271

Description:

1. Line 4239: Function zip calls function ct\_init, and crash happens
   1. Line 2297: The value of integer variable dist becomes 0
   2. Line 2300: a is a symbolic variable (assigned with 0 by SYM\_int)
      1. The value of expression (1<<(a-7)) becomes **33554432**
      2. Out of bound memory access happens at line 2301 because dist\_code only contains 512 elements
2. CROWN cannot generate more than 1 TCs as the first TC crashes.

4239. ct\_init(para1, para2);

/\* void ct\_init(ush \*para1, int \*para2){

\* … // dist is assigned with 16

\* 2297. dist >>= 7; // dist = 16>>7 = 0

\* // a and dist is not altered

\* 2300. for(int n = 0; n < (1<<(a-7)); n++){

\* 2301.  **dist\_code[256+dist++] = code;**  **// crash**

\* 2302. }

\* ...

\* }

\*/

Gzip-21

Function Name:file\_read

Reason: NSEF

Line No: 4289-4292

Description:

1. Line 4288: read is an external function, only the “then” branch is covered.
   1. Because CROWN cannot alter the external function read to cover the “else” branch (line 4289-4292)

4288. if((len = read(...)) == -1) {...}

**4289-4292 else{...} // not covered**

Gzip-22

Function Name: main

Reason: NSEF

Line No: 4665

Description:

1. Line 4662: proglen is assigned with 1 by the external function strlen
2. Line 4665: The “then branch” cannot be covered as proglen is less than 4

4662. **int** proglen = strlen(progname); // proglen = 1;

... // proglen is not altered

4665. **if**(proglen > 4 &&...) **{...} // not covered**

Gzip-23

Function Name: main

Reason: NSEF

Line No: 4674

Description:

1. Refer to Gzip-22 (the function signal is an external function)

Gzip-24

Function Name: main

Reason: NSEF

Line No: 4692-4697

Description:

1. Refer to Gzip-22 (the function strncmp is an external function)

Gzip-25

Function Name: main

Reason: NSEF

Line No: 4701-4746

Description:

1. Line 4701: variable opt becomes a concrete variable, which makes CROWN explore the “default branch” (line 4747) of “switch” statement only.
2. Other “case” branches cannot be covered

4701. **while**((optc = getopt\_long(...)) !=EOF){

4703. **switch** (optc){

**case 'b': ... break; // not covered**

**... // not covered**

4747. **default**: ... exit(1);

}

}

Gzip-26

Function Name: treat\_stdin

Reason: NSEF

Line No: 4791-4806

Description:

1. Line 4790: The external function isatty returns false
2. Line 4791-4800: The “then” branch of line 4790 cannot be covered because CROWN cannot make the external function return true.

4790. **if**(isatty(...))

**4791-4806.{... } // not covered**

Gzip-27

Function Name: treat\_stdin

Reason: NSEF

Line No: 4820

Description:

1. Line 4819: fstat is an external function and returns 0, which makes the “then” branch of line 4819 (i.e.,line 4820) unexecuted

4819. **if**(fstat(...) != 0)

**4820. {... } // not covered**

Gzip-28

Function Name: treat\_stdin

Reason: NSEF

Line No: 4832-4833

Description:

1. Line 4831: get\_method calls an external function (read) and terminates the execution, which makes line 4832-4833 unexecuted

4831. method = get\_method(); // execution is terminated

/\* int get\_method(){

\* ...

\* int len;

\* if((len = **read**(...)) == -1)) **exit(1)**;

\* …

\* }

\*/

**4832-4833. ... // not covered**

Gzip-29

Function Name: treat\_stdin

Reason: NSEF

Line No: 4842-4848

Description:Refer to Gzip-28

Gzip-30

Function Name: treat\_file

Reason: NSEF

Line No: 4870-4984

Description:

1. Line 4869: Function get\_istat calls an external function (stat) and returns **non-zero** value
   1. The “then branch” is explored and execution terminates
2. Line 4870-4984 cannot be covered

4869. **if**(get\_istat(...) != 0) **return**;

/\* int get\_istat(){

\* ...

\*

\* if(**stat(...) == 0**) return 0; // returns 0 in this case **only**

\* …

\* return -1;

\* }

\*/

**4870-4984 ... // not covered**

Gzip-31

Function Name: create\_outfile

Reason: NSEF

Line No: 5005-5007

Description:

1. Line 5004: function check\_ofname is called and it always returns 0, which makes the “then” branch of line 5004 unexecuted.

5004. **if**(check\_ofname(...) == -1)

/\* int check\_ofname(){

\* ...

\*

\* if(**stat(...) != 0**) return **0**; // stat always being executed

\* …

\* return -1; // not executed

\* }

\*/

**5005-5007{...} // not covered**

Gzip-32

Function Name: create\_outfile

Reason: NSEF

Line No: 5017-5051

Description:

1. Line 5010: An external function open is called and returns -1, which makes execution terminate at line 5015.
   1. Line 5017-5051 cannot be covered

5010. **int** ofd = open(ofname,...);

5011. **if**(ofd==-1){

...

5015. **return** -1;

5016. }

**5017-5051. ... // not covered**

Gzip-33

Function Name: get\_suffix

Reason: NSEF

Line No: 5088-5106

Description:

1. Line 5084: strrchr is an external function and returns NULL, which makes the execution terminates at line 5087
   1. Line 5088-5106 cannot be reached

5084. **char**\* p = strrchr(name, '.'); // return NULL

... // p is not altered

5087. **if**(p==NULL) **return** NULL; // terminated

**5088-5106. ... //not covered**

Gzip-34

Function Name: get\_istat

Reason: NSEF

Line No: 5130

Description: Refer to Gzip-30

Gzip-35

Function Name:get\_istat

Reason: NSEF

Line No: 5182-5188

Description:

1. Ref to Gzip-30

Gzip-36

Function Name: make\_ofname

Reason: NSEF

Line No: 5209-5224

Description:

1. Line 5200: function get\_suffix is executed and returns NULL , which makes the execution terminate at line 5203.
   1. Line 5209-5224 cannot be covered

5200. suff = get\_suffix(); // suff is assigned with NULL

... // suff is not altered

5203. **if**(suff==NULL) **return** -1;

**5209-5224. ... // not executed**

Gzip-37

Function Name: get\_method

Reason: NSEF

Line No: 5285-5357

Description:

1. Line 5283: function memcmp is an external function and returns non-zero
2. Line 5284-5357: The “then” branch of line 5283 cannot be covered.

5283. **if**(memcmp(...) == 0)

5284.

**5285-5357 {...} // not covered**

Gzip-38

Function Name: get\_method

Reason: NSEF

Line No: 5361-5370

Description:

1. Refer to Gzip-37

Gzip-39

Function Name: get\_method

Reason: NSEF

Line No: 5372-5373

Description:

1. Refer to Gzip-37

Gzip-40

Function Name: get\_method

Reason: NSEF

Line No: 5375-5378

Description:

1. Refer to Gzip-37

Gzip-41

Function Name: check\_ofname

Reason: NSEF

Line No: 5448-5494

Description:

1. Line 5447: stat is an external function and returns non-zero
   1. Execution exits at line 5447
2. Line 5448-5494 cannot be covered.

5447. **if**(stat(ofname,...)!=0) **return** 0;

**5448-5494. ... //not covered**

Gzip-42

Function Name:name\_too\_long

Reason: NSEF

Line No: 5423

Description:

1. Line 5423: stat is an external function, only the “else” branch is covered.
   1. Because CROWN cannot alter the external function stat to cover the “then” branch

5423. if(stat(...) == 0) {**... // not covered**}

else {...}

Gzip-43

Function Name:copy\_stat

Reason: NSEF

Line No: 5544

Description:

1. Line 5544: chmod is an external function
   1. Only the “then” branch is covered because CROWN cannot alter the external function chmod to cover the “else” branch

5544. if(chmod(...)){...}

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

Gzip-44

Function Name: \_\_getopt\_internal

Reason: NSEF

Line No: 5969

Description:

1. Line 5968:getenv is an external function and returns NULL, the “then branch” at line 5969 cannot be covered

5968.**if**(getenv (...) != NULL)

**5969. {...} // not covered**

Gzip-45

Function Name: \_\_getopt\_internal

Reason: NSEF

Line No: 6206-6244

Description:

1. Line 6188: my\_index is an **external function** which assigns **NULL** to variable temp
2. Line 6194: if **temp == NULL**, the “then” branch is explored
3. Line 6203. execution exits at 6204 inside the “then” branch of line 6194
4. Line 6206-6244 cannot be covered as CROWN cannot make the external function my\_index returns non-NULL value

6188. **char** \* temp = my\_index(...); // temp = NULL

... // temp is not altered

6194. **if**(temp==NULL)

{

...

6204. **return** '?';

}

**6206-6244. ... // not covered**