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 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 | [Grep-14](#i0n1el715mmg) | [Grep-18](#2gszpq5b5f4d) | [Grep-40](#tdxhfmyfztmr) | [Grep-41](#swf2ql2jpgw3) | [Grep-46](#t48cgbwoulpq) |  |  |
| NSEF | [Grep-3](#qhdo8ixsf3mn) | [Grep-4](#9cxp0sg7wwq6) | [Grep-7](#rh1splz5ef2a) | [Grep-9](#rjjngcwb21ap) | [Grep-12](#jx8xq5dw4fwb) | [Grep-19](#4a9yvsnhzyv3) | [Grep-20](#kme85nxhft71) |
| [Grep-23](#41haaftjrvvl) | [Grep-29](#v0iqsontjzvm) | [Grep-33](#fhfmo7f8rmjz) | [Grep-42](#f4a4so3jss80) |  |  |  |
| NSFF | [Grep-5](#ql130wuc2qyu) |  |  |  |  |  |  |
| NSGP | [Grep-44](#fuqpoim12jb) |  |  |  |  |  |  |
| NSLS | [Grep-2](#vuma2abl8yf7) | [Grep-25](#by946e42msza) | [Grep-26](#80itv6wpcqzm) | [Grep-27](#y4fnb8aryhn4) | [Grep-45](#19af3e4zexlq) |  |  |
| NSSP | [Grep-1](#i4soqre4e40m) | [Grep-6](#z81bljdi5kw3) | [Grep-8](#ev1pu5mo4zfy) | [Grep-10](#1hjmrpkbn7aw) | [Grep-13](#sm4tvxypyyl7) | [Grep-28](#thzq71hd9to0) | [Grep-30](#mmsi1kk0lrtw) |
| [Grep-31](#kx7k2yvvgokd) | [Grep-32](#2daxc2bf5qh6) | [Grep-34](#gja9ebtppdm) | [Grep-35](#ywk36lkxwhew) | [Grep-36](#rrf62u1xplk3) | [Grep-43](#gzgodlmmet8p) |  |
| RT1 | [Grep-21](#do7rky5snxji) | [Grep-24](#lsroq8read4g) |  |  |  |  |  |
| TIWC | [Grep-14](#i0n1el715mmg) | [Grep-15](#8b7wb6pavthq) | [Grep-18](#2gszpq5b5f4d) |  |  |  |  |
| UB | [Grep-3](#qhdo8ixsf3mn) | [Grep-11](#omckr5hfgacm) | [Grep-19](#4a9yvsnhzyv3) | [Grep-20](#kme85nxhft71) |  |  |  |
| USV | [Grep-16](#lm82vvg7pjhs) | [Grep-17](#8tlgoo5qxgt1) | [Grep-22](#yk65yzuegfhf) | [Grep-37](#nr70oti3s4t1) | [Grep-38](#upjhqq7s5k7o) | [Grep-39](#41m43s3da51v) |  |

Grep-1

Function Name: xrealloc

Reason: NSSP

Line No: 191

Description:

1. Line 182: Variable ptr is a symbolic pointer, and CROWN always allocates 1 byte for this pointer (i.e., ptr = malloc(1\*sizeof(char)).
2. Line 191: The “else” branch of “if” statement cannot be covered as CROWN cannot alter ptr to NULL pointer.

182. **char** \* xrealloc(**char** \*ptr,...){

... // ptr is not modified

188. **if**(ptr) {...}

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

...

195. }

Grep-2

Function Name: reset

Reason: NSLS

Line No: 239

Description:

1. Line 237: variable init is a local static variable (assigned with 0)
2. Line 239: the “else” branch of “if” statement cannot be covered as the value of init is always 0 and CROWN cannot declare symbolic local static variables.

234. void reset(){ // begin of function reset

…

237. **static** **int** init = 0;

238. // init is not modified

239. **if**(!init){...}

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

**…**

268. } // end of function reset

Grep-3

Function Name: reset

Reason: NSEF, UB (probably)

Line No: 253

Description:

1. Line 251: buf is allocated with 40K bytes
2. Line 252: buf is always non-NULL as the machine used by CR2 has 16G (>>40K) bytes
   1. The “then” branch of “if” statement (i.e., line 253) cannot be covered

251. **char** \* buf = malloc(8192\*5 + 1);

252. **if**(!buf)

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

Grep-4

Function Name: reset

Reason: NSEF

Line No: 261-265

Description:

1. Line 259: Function fstat is an external function and it returns a negative value (<0).
   1. The “else” branch of “if” statement (i.e., 261-265) cannot be covered as CROWN cannot make fstat return a non-negative value (>=0)

259. **if**(fstat(...) < 0)

260. {...}

**261-265. else {...} // not covered**

Grep-5

Function Name: prline

Reason: NSEF

Line No: 403

Description:

1. Line 402: external function ferror returns 0 so that the “then” branch (i.e., line 403) cannot be covered
   1. Because CROWN cannot make the file-handling external function return non-zero value

402. **if**(ferror(stdout))

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

Grep-6

Function Name: prpending

Reason: NSSP

Line No: 415

Description:

1. Line 414: lastout is a symbolic pointer and it is allocated with 1 byte by CROWN
   1. The “then” branch (i.e., line 415) cannot be covered as CROWN cannot alter symbolic pointer lastout to NULL

… // lastout is not modified

414. **if**(lastout == NULL) // lastout = malloc(1\*sizeof(char))

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

Grep-7

Function Name: prpending

Reason: NSEF

Line No: 421

Description:

1. Line 420: Function memchr is an external function and it returns 0
   1. The “then” branch of “if” statement (i.e., 421) cannot be covered as CROWN cannot make memchr return non-zero value

420. **if**(memchr(...) != 0)

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

Grep-8

Function Name: prtext

Reason: NSSP

Line No: 451-453

Description:

1. Line 450: both p and bp are symbolic character pointer (char \*) and p < bp.
   1. CROWN cannot alter the value of symbolic pointers to satisfy the condition (p > bp) so that the “then” branch (line 451-453) cannot be covered

450. **if**(p > bp)

**451-453 {...} // then branch is not covered**

Grep-9

Function Name: main

Reason: NSEF

Line No: 474

Description:

1. Line 473: Function memchr is an external function and it returns 0
   1. The “then” branch of “if” statement (i.e., 474) cannot be covered as CROWN cannot make memchr return non-zero value

473. **if**(memchr(...) != 0)

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

Grep-10

Function Name: grepbuf

Reason: NSSP

Line No: 506-521

Description:

1. Line 505: both p and end are symbolic character pointer (char \*) and p < end.
   1. CROWN cannot alter the value of symbolic pointers to satisfy the condition (p > end) so that the “then” branch (line 451-453) cannot be covered

505. **while**(p > end)

**506-520. {...} // then branch is not covered**

Grep-11

Function Name: grep

Reason: UB

Line No: 562-587

Description:

1. Line 550: A “for” loop is executed.
2. Line 552: bufbeg, buffer, and buflim are pointer variable
3. Line 560: the condition of the “if” statement is always satisfied (see line 552), and the “for” loop is finished.
   1. The remaining part of the “for” loop (line 561-587) cannot be covered.

550. **for**(;;){ // beginning of for loop

... // bufsalloc is not used by other conditions

552. bufbeg= **buffer** + bufsalloc - save, buflim = **buffer+**bufsalloc; // fillbuf()

... // none of bufbeg, buflim, bufsalloc, and save are altered

560. **if**(buflim - bufbeg == save)

561.  **break**;

**562-587 ... // not covered**

} // end of for loop

Grep-12

Function Name: main

Reason: NSEF

Line No: 657-786

Description:

1. Line 656: getopt\_long is an external function and the execution of this function returns EOF, the “then” branch (i.e., line 657-786) cannot be covered by CROWN

656. **while**((opt=getopt\_long(...)) != EOF){

**... // not covered**

786. }

Grep-13

Function Name: regex\_compile

Reason: NSSP

Line No: 2827-3682

Description:

1. Line 2734: const char\* p = pattern, \*pend = pattern+size
2. size is symbolic integer variable and its initial value is 0
   1. So p==pend
3. Line 2826: the “then” branch of this line cannot be covered as CR2 cannot alter p and pend to satisfy p!=pend

2734. **const** **char**\* p = pattern, \*pend = pattern+size;

... // p and pend is not altered

2826. **while**(p!=pend)

**2827-3682{...} // not covered**

... // p and pend is not altered

Grep-14

Function Name: re\_compile\_fastmap

Reason: TIWC, FTCC

Line No: 4183-4451 (whole re\_compile\_fastmap function body)

Description: TC itself does not cause crash, but crash happens when writing symbolic information (e.g., SPF) into disk (i.e., file szd\_execution) due to the memory conflict between CROWN engine and TC.

1. Error Message:
   1. Program received signal SIGSEGV, Segmentation fault.

0x000000000054e1b0 in crown::SymbolicExecutionWriter::Serialize (this=0x58dce8, os=..., tracker=0x58dc20)

at libcrown/symbolic\_execution\_writer.cc:69

69 exprs\_[i->first]->Serialize(os);

Grep-15

Function Name: re\_search\_2

Reason: TIWC

Line No: 4577-4641

Description: Refer to Grep-14

Grep-16

Function Name: re\_match\_2

Reason: USV

Line No: 5187-5936

Description:

1. Line 5174: \*p is a symbolic integer, and its initial value is 0;
   1. TC1: case ‘0’ is reached, and execution exits normally.

TC2: the case “default” is reached, and crash happens (as function abort is called)

* + 1. other “case” statements of the “switch” statement cannot be covered

5363. int re\_match\_2(...){

…

5174. **switch** (\*p++){ // \*p is a symbolic integer

**case** 0: ... **break**; // execution of TC1

**5187. case 1: ... break; // not covered**

**case 2; ... break; // not covered**

**... // not covered**

**default**: **abort();** // execution of TC2, crash happens

5936. }

…

return -1

}

Grep-17

Function Name: re\_match\_2

Reason: USV

Line No: 5040-5061

Description: Refer to Grep-16

Grep-18

Function Name: regexec

Reason: TIWC, FTCC

Line No: 6467-6525

Description: First TC itself does not crash, but CROWN engine crashes when it tries to write the symbolic information (e.g., SPF) into the physical files (i.e., szd\_execution)

Error message: 0x00007ffff74a2c82 in \_\_GI\_\_\_assert\_fail (assertion=0x562e51 "stack\_.size() > 0",

file=0x562e30 "libcrown/symbolic\_interpreter.cc", line=140

Grep-19

Function Name: xcalloc

Reason: NSEF, UB

Line No: 6719

Description:

1. Line 6716: n is a symbolic integer and its initial value is 0.
2. Line 6716: Function malloc(n) returns a pointer which points to the first address of n bytes.
   1. If the value of n is 0, a **non-null** pointer would be returned by malloc [(ref)](https://man7.org/linux/man-pages/man3/malloc.3.html#:~:text=then%20malloc()%20returns%20either%20NULL%2C%20or%20a%20unique%20pointer%0A%20%20%20%20%20%20%20value%20that%20can%20later%20be%20successfully%20passed%20to%20free().)
3. Line 6716: r is assigned with the return value of malloc (i.e., non-NULL), so the “then” branch of “if” statement at line 6718 cannot be covered.

... // n is not used by other branch conditions

6716. **void** \* r = malloc(n);

6717. // r is not altered

6718. **if**(r==NULL)

**6719 {...} // not covered**

... // n is not used by other branch conditions

Grep-20

Function Name: xmalloc\_1

Reason: NSEF, UB

Line No: 6731

Description: Refer to [Grep-19](#4a9yvsnhzyv3)

Grep-21

Function Name: charclass\_index

Reason: RT1

Line No: 6880-6886

Description: Timeout1 for the first TC is reached so that no more TCs can be generated

1. Line 6883: a and b are symbolic variables (assigned with 0 for the first TC)
2. Line 6883: the “while” loop is an infinite loop as a=b=0
   1. The first TC executes this line (infinite loop), and then timeout1 is reached

**6883. if(a >= b) {while(a>=b) b=b\*2;} // not covered by gcov**

Grep-22

Function Name: lex

Reason: USV

Line No: 7196-7274

Description:

1. Line 7196: x is a symbolic variable, and CROWN generates 1 TC which assigns '[' to x to explore the “then” branch of “if” statement
2. Line 7273: the execution of current TC has an infinite “while” loop as a=b=0 (initial value of symbolic variable is 0)
   1. Timeout1 is reached, and execution is killed, so that gcov cannot record the coverage of current TC

**7196. if(x=='['){ // not covered begin**

**... // a == b = 0, and a and b are not altered**

**7273. while(a>=b) b\*=2;**

**7274. } // not covered end**

Grep-23

Function Name: lex

Reason: NSEF

Line No: 7105-7114

Description:

1. Line 7105: isdigit is an external function and the “else” branch is covered.
   1. CROWN can not alter the external function to cover the “then” branch

**7105. if(isdigit(...)){ // not covered begin**

**...**

**7114. } // not covered end**

Grep-24

Function Name: state\_index

Reason: RT1

Line No: 7639-7648

Description: Refer to [Grep-21](#do7rky5snxji)

Grep-25

Function Name: dfastate

Reason: NSLS

Line No: 8095

Description:

1. Line 8091: variable init is a local static variable (assigned with 0)
2. Line 8095: the “else” branch of “if” statement cannot be covered as the value of init is always 0 and CROWN cannot declare symbolic local static variables.

8067. **void** dfastate(...){

...

8091. **static** **int** init = 0;

...// init is not altered

8095. **if**(!init){...}

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

...

8306. }

Grep-26

Function Name: dfaexec

Reason: NSLS

Line No: 8478-8485

Description: Refer to [Grep-25](#by946e42msza)

Grep-27

Function Name: dfaexec

Reason: NSLS

Line No: 8436

Description: Refer to [Grep-25](#by946e42msza)

Grep-28

Function Name: icatalloc

Reason: NSSP

Line No: 8701

Description: CROWN always allocates 1 byte function parameter of character pointer type (malloc (1\*sizeof(char))), and CROWN cannot alter the pointer to NULL.

1. Line 8694: variable new is a symbolic non-NULL pointer.
2. Line 8701: new is non-NULL, so the “then” branch cannot be covered

8694. **char** \* icatalloc(**char**\* **new**,...){ //new=malloc(1\*sizeof(char))

...// new is not altered

8701. **if**(new=NULL)

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

...

}

Grep-29

Function Name: istrstr

Reason: NSEF

Line No: 8733

Description:

1. Line 8733: strncmp is an external function, and it returns 0. CROWN cannot make the external function return non-zero value.
   1. The “else” branch cannot be covered.

8733. **if** (strncmp (...) == 0){...}

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

Grep-30

Function Name: ifree

Reason: NSSP

Line No: 8742

Description: CROWN always allocates 1 byte function parameter of character pointer type (malloc (1\*sizeof(char))), and CROWN cannot alter the pointer to NULL.

1. Line 8739: variable cp is a symbolic non-NULL pointer.
2. Line 8742: the “else” branch cannot be covered because cp is not NULL

8739. **char** \* ifree(**char**\* **cp**){ //cp=malloc(1\*sizeof(char))

...// cp is not altered

8742. **if**(**cp**!=NULL) {...}

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

}

Grep-31

Function Name: freelist

Reason: NSSP

Line No: 8753

Description: CROWN always allocates 1 byte function parameter of character pointer type (malloc (1\*sizeof(char))), and CROWN cannot alter the pointer to NULL.

1. Line 8747: variable cpp is a symbolic non-NULL pointer.
2. Line 8753: the “then” branch cannot be covered because cpp is not NULL

8747. **char** \* freelist(**char**\* **cpp**){ //cpp=malloc(1\*sizeof(char))

...// cpp is not altered

8753. **if**(**cpp**==NULL)

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

}

Grep-32

Function Name: comsubs

Reason: NSSP

Line No: 8820, 8823

Description: Refer to [Grep-31](#kx7k2yvvgokd)

Grep-33

Function Name: comsubs

Reason: NSEF

Line No: 8830-8836

Description:

1. line 8829: strstr is an external function and the execution of this function returns NULL.
   1. CROWN cannot make this external function returns non-NULL value, so the “then” branch (line 8830-8836) cannot be covered

8829. **while**(strchr(...) !=NULL) // macro “index”

**8830-8836. {...} // not covered**

Grep-34

Function Name: addlists

Reason: NSSP

Line No: 8853

Description: CROWN always allocates 1 element for the parameter of pointer type.

1. Line 8846: both old and new are non-NULL pointers
2. Line 8852: the “then” branch cannot be covered

8846. **char** \*\* addlists(**char**\*\* old, **char**\*\* **new**){

...// new and old are not altered

8852. **if**(**new**==NULL || old == NULL)

**8853. {...} // not covered**

...

}

Grep-35

Function Name: inboth

Reason: NSSP

Line No: 8875

Description: Refer to [Grep-34](#gja9ebtppdm)

Grep-36

Function Name: dfamust

Reason: NSSP

Line No: 8977

Description:

1. Line 8977: both mp and musts are pointers and mp<=musts is satisfied
2. Line 8977: CROWN cannot alter the value of pointer mp or musts to explore the “else” branch of “if” statement

8977.**if**(mp<=musts){...} // mp = char\*, musts=char\*

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

Grep-37

Function Name: kwsincr

Reason: USV

Line No: 9336-9450

Description:

1. Line 9336: len is a symbolic unsigned integer. To explore the then branch of “while” statement, CROWN outputs a very large value (=0xfffffffffffffffd) and assigns it to variable len
2. Line 9338: an out-of-bound memory access occurs as text is allocated with one element only, but (\*text++) is executed more than 1 time.
   1. As the execution crashes, gcov cannot record the coverage of the “then” branch of “while” statement of line 9336

**9336. while(len--){ // not covered begin**

**...**

**9338. char label = \*(text++); // crash , as text = char[1]**

**...**

**} // not covered end**

Grep-38

Function Name: bmexec

Reason: USV

Line No: 9708-9772

Description: CROWN outputs unrealistic value for symbolic variables , and then crash happens.

1. Line 9706: len and size are symbolic variables. To cover the “else” branch of “if” statement, CROWN outputs **2^31-1** and assigns it to both len and size
2. Line 9713: **crash** happens as CROWN allocates only one element for the pointer variable text (i.e., text = malloc(1\*sizeof(int)))
   1. But this line tries to access text[2^31-3] (out of bound memory access)
   2. The coverage of “else” branch (i.e., line 9708-9772) cannot be recorded by gcov.

9706. **if**(len > size)

9707. **return** 0； // len = 1, size = 0

**9708. else{ // not covered begin**

**... // len is not altered**

**9713. int gc = \*(text + len - 2) // equivalent with text[2^31-3], crash**

**9772. } // not covered end**

Grep-39

Function Name: cwexec

Reason: USV

Line No: 9875-9919

Description:

1. Line 9799: lim , mch, and end points to the same address as the value of symbolic variable size is 0, and size is not used by other branch conditions.
2. Line 9870:a is a symbolic integer variable
   1. To cover the “then” branch of “if” statement, CROWN outputs a negative number **-2147483648** and assigns it to a
      1. In this case, lim - end = -2147483648 (< 1), the “then” branch of line 9874 cannot be covered.
   2. If the “else” branch is covered, a is assigned with 0
      1. In this case, lim - end = 0 (< 1), the “then” branch of line 9874 cannot be covered

9799. **char** \* lim= text, \*end = text+size, \*mch = text;

... // lim, end, and mch are not altered

... // a is not used by other branch conditions

9870. **if**(lim - mch > a) // lim-mch == 0

9871. lim = mch + a;

... // lim, end, and mch are not altered

9874. **while**(lim - end >= 1)

**9875-9919{...} // not covered**

Grep-40

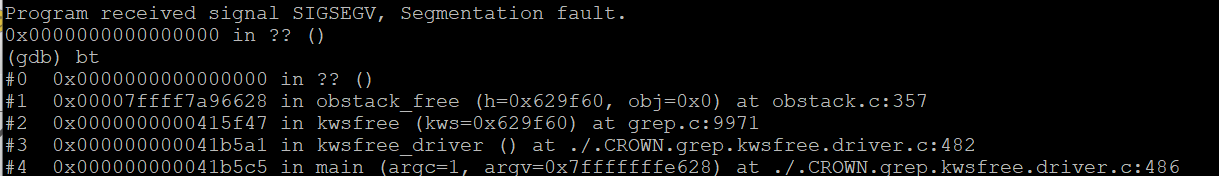
Function Name: kwsfree

Reason: FTCC

Line No: 9965-9973

Description: The first TC crashes, which makes CROWN not able to generate more than 1 TCs

1. Line 9971: The first TC executes the external library function (obstack\_free) and crashes.
   1. The documentation of this function is not available [<ref>](https://man7.org/linux/man-pages/man3/undocumented.3.html#:~:text=3)%2C%0A%20%20%20%20%20%20%20nlist(3)%2C-,obstack_free(3),-%2C%20parse_printf_format(3)%2C%20p_cdname) , and the reason why this external function crashes is unknown.
   2. Debug trace:



**9965. void kwsfree(){ // not covered begin**

**...**

**9971. obstack\_free(...); // crash happens**

**...**

**9973. } // not covered end**

Grep-41

Function Name: kwsmusts

Reason: FTCC

Line No: 10586-10617

Description: The first TC crashes, which makes CROWN not able to generate more than 1 TCs

1. Line 10602: crash happens because this line tries to dereference a NULL pointer (d->next->next->must)

kwsmust\_driver(){

...

d->next->next = malloc(...);

d->next->next->must = **NULL**;

...

kwsmust();

}

**10586. void kwsmust(){ // not covered begin**

**...**

**10607. for(dm = d; dm; dm=dm->next){**

**...**

**10611. char a = \*(d->must); // crash happens at the**

**// third execution line 10611**

**...**

**10613. }**

**...**

**10617. } // not covered end**

Grep-42

Function Name: Ecompile

Reason: NSEF

Line No: 10688-10691

Description:

1. Line 10687: strcmp is an external function and returns non-zero, only the “else” branch is covered.
   1. CROWN cannot make the external function return 0, the “then” branch (line 10688-10691) is not covered.

10687. **if**(strcmp(...)==0)

**10688-10691. {...} // not covered**

Grep-43

Function Name: Ecompile

Reason: NSSP

Line No: 10700-10741

Description:

1. Line 10698: msg is a pointer, and CROWN allocates one byte for this pointer (msg = malloc(1\*sizeof(char)));
2. Line 10698: CROWN cannot alter msg to NULL, so the “else” branch cannot be covered.

...// msg is not altered

10698. **if**(msg != NULL) {exit(0);} // call re\_compile\_pattern()

**10700-10741. else{...} // not covered**

Grep-44

Function Name: EGexecute

Reason: NSGP

Line No: 10760-10785

Description:

1. Line 10759: kwset is a global pointer of (void \*) type, and CROWN just assigns NULL to kwset
2. Line 10759：the “then” branch cannot be covered.

… // kwset is not altered

10759. **if**(kwset)

**10760-10784. {...} // not covered**

Grep-45

Function Name: EGexecute

Reason: NSLS

Line No: 10794-10843

Description:

1. Line 10791: init is a local static variable (assigned with 0)
   1. The “else” branch (line 10794-10843) cannot be covered as CROWN can not declare local static variables as symbolic variables.

char \* EGexecute(...){

static int init = 0;

...// init is not altered

10791. **if**(init == 0) {

…

10793. }

**10794-10843. else{...} // not covered**

}

Grep-46

Function Name: Fexecute

Reason: FTCC

Line No: 10879-10929

Description:The first TC crashes, which makes CROWN not able to generate more than 1 TCs

1. kwset is a global (void \*) pointer, and CROWN assigns NULL to kwset
2. The execution of the first TC executes line 10890, where a dereference on NULL pointer happens
   1. No more TCs can be generated

… // kwset is not altered

10890. int len = kwset -> mind // crash as kwset is NULL