ERICSSON

TITAN 6.3.1 CHANGE LOG

2017.12.01

BUG 521410 - ASN.1 NAMED BIT SEMANTIC CHECK RESULTS FALSE ERROR

- IEEE Security definitions ASN.1
- ETSI is changing ITS security definitions to ASN.1, this problem come up during this work.
- In ASN.1, named bits identify the position of the bit(s) within the instance, which shall be set to 1. All other bits in the instance are set to 0. But this doesn't influence the value space of the type.
- For example in
- EndEntityType ::= BIT STRING {app (0), enrol (1) } (SIZE (8)) (ALL EXCEPT {})
- the actual value of app is: '10000000'B and the value of enroll is: '01000000'B
- (and {app,enroll} would be '11000000'B)
- Titan, when semantic checking the _use_ (see attached file) of the above named bit 'app', gives the false error:
- ../asn1/LibIts/Security/IEEE1609dot2.asn:293: error: '1'B is not a valid value for type `bitstring
- Named bit are specified in X.680(2008, 2015) clause 22.
- Fixed.
- Named bits are now extended (with zeros) to reach the minimum length restriction.

BUG 527623 - IMPLEMENT VERDICT REDIRECT FOR 'DONE' STATEMENT

- The 'done' statement's value redirect is different in TITAN than in the TTCN-3 standard.
- In TITAN it stores the behavior function's return value. It can only be used if the behavior function returns a value. The type of the variable in the value redirect must be the same as (or compatible with) the type returned by the behavior function. This type must also have the attribute 'extension "done"'. There must also be a matching template (of the same type as the return value) in the 'done' statement if the value redirect is used.
- According to the TTCN-3 standard the 'done' statement's value redirect should store the local verdict on the component, after the behavior function has finished executing. Also, the 'done' statement cannot contain a matching template.
- The standard compliant behavior should be added to TITAN in the following way:
- - if the 'done' statement contains a matching template, then the value redirect works as before (it stores the function's return value), and
- - if the 'done' statement doesn't contain a matching template, then the value redirect stores the component's local verdict (as per the standard).
- Change implemented.
- Regression tests added under regression_test/done.

BUG 527653 - IMPLEMENT EXTENDABLE SEQUENCE CODING IN OER

BUG 527793 - TEXT CODEC BUG

- I'm running a TEXT based MGCP decoder in a somewhat more complex parallel test configuration, and it fails with the following backtrace:

- Code: [Select all] [Show/ hide]
- ./IPA_Test: Abort was called
- /usr/lib/titan/libttcn3-parallel-dynamic.so(_Z14signal_handleri+0xa3)[0x7f9a77b23813]
- /lib/x86_64-linux-gnu/libc.so.6(+0x33af0)[0x7f9a761e7af0]
- /lib/x86_64-linux-gnu/libc.so.6(gsignal+0x110)[0x7f9a761e7a70]
- /lib/x86_64-linux-gnu/libc.so.6(abort+0x17a)[0x7f9a761e919a]
- /lib/x86_64-linux-gnu/libc.so.6(regerror+0x75)[0x7f9a76286985]
- /usr/lib/titan/libttcn3-parallel-dynamic.so(_ZNK11Token_Match11match_beginER11TTCN_Buffer+0x1b1)[0x7f9a77af1b11]
- ./MGCP_Types.so(_ZN11MGCP__Types11MgcpCommand11TEXT_decodeERK21TTCN_Typedescriptor_tR11TTCN_BufferR16Limit_Token_Listbb+0x90)[0x7f9a7ea70904]
- ./MGCP_Types.so(_ZN11MGCP__Types11MgcpCommand6decodeERK21TTCN_Typedescriptor_tR11TTCN_BufferN11TTCN_EncDec8coding_tEz+0x417)[0x7f9a7ea70229]
- ./MGCP_Types.so(_ZN11MGCP__Types16dec__MgcpCommandERK10CHARSTRING+0xbc)[0x7f9a7ea81323]
- ./IPA_Emulation.so(_ZN14IPA__Emulation17f__mgcp__to__userERK11OCTETSTRING+0x11b)[0x7f9a7fa0ad41]
- ./IPA_Emulation.so(_ZN14IPA__Emulation10ScanEventsEv+0x520)[0x7f9a7fa0b44d]
- ./IPA_Emulation.so(_ZN14IPA__Emulation12main__clientERK10CHARSTRINGRK7INTEGERS2_S5_+0xa1)[0x7f9a7fa0a8c4]
- ./IPA_Emulation.so(+0x21ba5)[0x7f9a7fa0cba5]
- /usr/lib/titan/libttcn3-parallel-dynamic.so(_ZN11Module_List14start_functionEPKcS1_R8Text_Buf+0x2b)[0x7f9a77ad5d3b]
- /usr/lib/titan/libttcn3-parallel-dynamic.so(_ZN12TTCN_Runtime14start_functionEPKcS1_R8Text_Buf+0x25)[0x7f9a77ae69e5]
- /usr/lib/titan/libttcn3-parallel-dynamic.so(_ZN18TTCN_Communication13process_startEv+0x42)[0x7f9a77aad802]
- /usr/lib/titan/libttcn3-parallel-dynamic.so(_ZN18TTCN_Communication23process_all_messages_tcEv+0x2f5)[0x7f9a77aae055]
- /usr/lib/titan/libttcn3-parallel-dynamic.so(_ZN12TTCN_Runtime8ptc_mainEv+0xdf)[0x7f9a77aea50f]
- /usr/lib/titan/libttcn3-parallel-dynamic.so(main+0x330)[0x7f9a77824da0]
- /lib/x86_64-linux-gnu/libc.so.6(__libc_start_main+0xf1)[0x7f9a761d4561]
- ./IPA_Test(+0xa1a)[0x564e293a0a1a]
- MC@nataraja: Unexpected end of PTC connection (6) from 127.0.0.1 [127.0.0.1].

- So the generated code wants to do something with regular expressions and then that fails, hence it calls regerror() but this in turn crashes with an abort.

- In TEXT.cc of titan.core I can find
- Code: [Select all] [Show/ hide]
- int Token_Match::match_begin(TTCN_Buffer& buff) const
- {
- int retval=-1;
- int ret_val=-1;
- if(null_match){
- if (TTCN_EncDec::get_error_behavior(TTCN_EncDec::ET_LOG_MATCHING) !=
- TTCN_EncDec::EB_IGNORE) {
- char msg[ERRMSG_BUFSIZE2];
- regerror(ret_val, &posix_regexp_begin, msg, ERRMSG_BUFSIZE2);

BUG 527793 - TEXT CODEC BUG

- Comment from Gabor Szalai:

- "
- The
- char msg[ERRMSG_BUFSIZE2];
- regerror(ret_val, &posix_regexp_begin, msg, ERRMSG_BUFSIZE2);

- shouldn't be there.
- “

- Faulty regerror() call removed.

BUG 527847 - STR2FLOAT SHOULD HANDLE SPECIAL FLOAT VALUES

- Currently it does not handle the special float values like -+ infinity and not_a_number
- Changed str2float() and float2str() to handle the special float values. The string equivalents for these values in both functions are: "infinity", "-infinity" and "not_a_number".

- Regression tests added under regression_test/predefFunction.

ERICSSON