Early Results from

Characterizing Verification Tools through Coding Error Candidates Reported in Space Flight Software

R. Gerlich[1], R. Gerlich[1], A. Fischer[2] , M. Pinto[2] , C.R. Prause[3]

Data Systems in Aerospace DASIA 2016

May 12th, 2016, Tallinn, Estonia

[1] Dr. Rainer Gerlich BSSE System and Software Engineering
Immenstaad, Germany
E-Mail: Rainer.Gerlich@bsse.biz
Ralf.Gerlich@bsse.biz

[3] Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)
Bonn, Germany
E-Mail: Christian.Prause@dlr.de

[2] etamax space GmbH
Braunschweig, Germany
E-Mail: A.Fischer@etamax.de
M.Pinto@etamax.de

Contents

- Introduction
- Terms and Definitions
- Tool Characterization
- Lessons Learned
- Results
- Conclusions/Outlook

Introduction

Note

The results presented here **strongly depend** on

- putting focus on safety critical issues

  tool messages must address faults which result in a failure or in a violation of good engineering practices

- the chosen application software and its fault profile

  some fault types may not be present

- the selected subset of functions subject of evaluation

  some fault types may not be present in this subset

- the observed number of defects per defect type

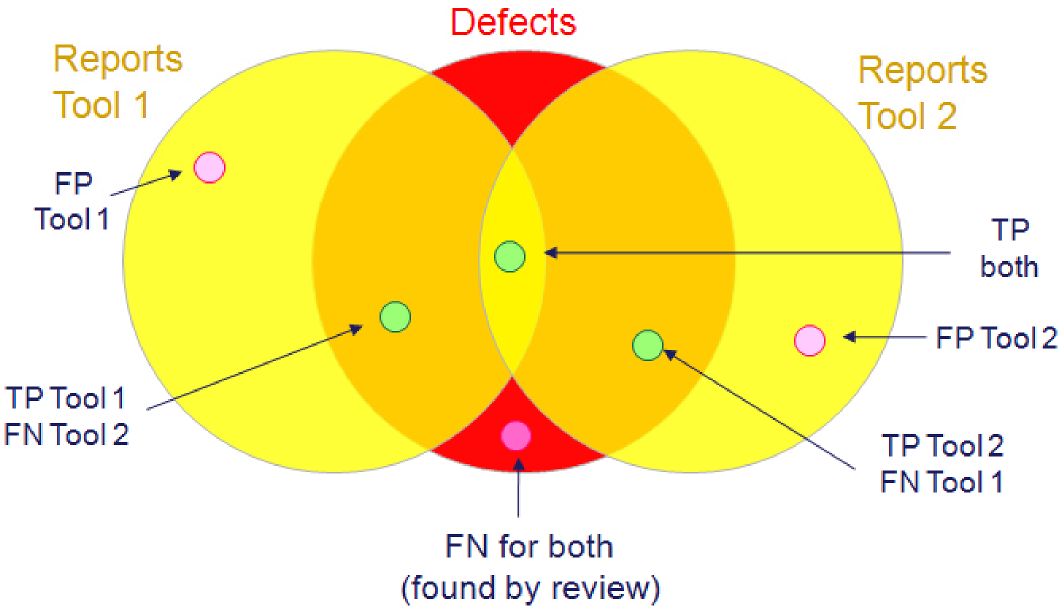
  the number of defects acts as a weight when deriving figures over all defect types

Terms and Definitions

Defect Terminology

Term	Scope
Fault	Mistake in code
⇓	⇓
Error	Bad state of a system
⇓	⇓
Failure	Unexpected observed behaviour

- A *defect* commonly refers to troubles with a software product, with its external behavior or its internal features (e.g., its maintainability).
- This includes consideration of the risk of faults by potential changes of the context putting focus on safety critical issues.



		Code	
		Defect present	*Defect NOT present*
Result	*Defect Reported*	true positive	false positive
	Defect NOT reported	false negative	true negative

List of Considered Defect Types

Defect Type	Criticality Level
Array Index Out-of-Bounds	Critical
Dereference of Invalid Pointer	Critical
Dereference of NULL-Pointer	Critical
File Access Error	Critical
Invalid function pointer	Critical
Non-terminating Loop	Critical
Passing invalid argument to standard library routine	Critical
(Possible) Recursion	Critical
Resource Leak	Critical
Undefined Result of Arithmetic Operation	Critical
Uninitialized Variable	Critical
Arithmetic Operation on NULL Pointer	Warning
Invariant Condition	Warning
Invariant Expression	Warning
Parameter Type Mismatch in Function Call	Warning
Timeout during execution	Warning
Unnecessary loop construct	Warning
Unreachable Code	Warning
Unused Result	Warning
Multiple return paths	Uncritical

Software and Tool Characterization

Issued Reports

Tool	1	2	3	4	5	6
Reports	146	742	1481	4995	2106	9

- Initially ~ 95.000 reports

  mainly lexical violations

  deactivated by configuration

- *If rules are not considered right from the beginning, many reports at the end*

- too many non-important reports resulting in missed important reports

- false negatives of the tool !

- *happened in case of two tools*

- *relevant messages were reported in a group of messages considered as negligible according to the naming convention of the tool and the presence of thousands of reports*

- *recognized at the end when looking carefully at the summary figures*

  searching for the reason of false negatives for this tool

Comparison of Reports

- **Comparison**
  - quite different from considering a report from a single tool
  - reports for the same defect do differ

- **Issues**
  - printed reports may not provide line numbers
  - same defect may be reported at different lines and even in different files
  - a defect may be reported more than once
  - classification as TP or FP may depend on context

    context: the history of execution until the location of the defect is reached, i.e. the function is not called on top-level

    w/o context: function is called on top-level
  - reports may be grouped or not

The Software

Defect Type	True Pos		Evaluated Subset “weighted”
	with ctxt	w/o ctxt	
Array Index Out-of-Bounds	120	126	🟥 x
Dereference of Invalid Pointer	31	47	🟥 x
Dereference of NULL-Pointer	3	8	🟥 x
File Access Error	1	1	🟥 x
Invalid function pointer	2	2	🟥 x
Non-terminating Loop	1	1	🟥 x
Passing invalid argument to standard library routine	1	1	🟥 x
(Possible) Recursion	1	1	🟥 x
Resource Leak	2	2	🟥 x
Undefined Result of Arithmetic Operation	2	2	🟥 x
Uninitialized Variable	14	15	🟥 x
Arithmetic Operation on NULL Pointer	0	3	
Invariant Condition	16	12	🟧 x
Invariant Expression	44	44	
Parameter Type Mismatch in Function Call	2	2	🟧 x
Timeout during execution	1	2	🟧 x
Unnecessary loop construct	1	1	
Unreachable Code	61	45	🟧 x
Unused Result	58	58	🟧 x
Multiple return paths	12	12	
Total, 60 functions	**373**	**385**	**FP ~ 25%**
Reports in total, 610 functions: 150 .. 5000			

Total	
Property	**Quantity**
Size / KLOC	42
Functions, total	610
API Functions	376
c-Files	39
h-files	96

Evaluated 6 Tools, 4 Combinations each	
Property	**Quantity**
Size / KLOC	4
Functions, random	31
Functions, by fault distribution	29
Functions, total	60
c-Files	24

Defect Statistics

Report Grouping		Number of Reports				
		TP+FP	TP		FP	
			with ctxt	w/o ctxt	with ctxt	w/o ctxt
Non-Grouped	all defect types	500	369	381	131	119
	weighted	439	311	320	129	119
Grouped	all defect types	270	195	201	75	69
	weighted	231	159	162	72	69

Coverage of Defect Types

Defect Type		TP Reported by Tool						
		1	2	3	4	5	6	Supporting Tools
Array Index Out-of-Bounds		x	x	x	x	x		5
Dereference of Invalid Pointer		x	x	x		x		4
Dereference of NULL-Pointer			x	x		x		3
File Access Error				x				1
Invalid function pointer			x					1
Non-terminating Loop			x			x		2
Passing invalid argument to standard library routine			x					1
(Possible) Recursion		+	+			x		3
Resource Leak					x			1
Undefined Result of Arithmetic Operation		x	x			x		3
Uninitialized Variable		x	x		x	x		4
Arithmetic Operation on NULL Pointer						x		1
Invariant Condition			x	x	x	x		4
Invariant Expression						x		1
Parameter Type Mismatch in Function Call							x	1
Timeout during execution				x				1
Unnecessary loop construct						x		1
Unreachable Code		x	x	x	x	x		5
Unused Result		x			+	x		3
Multiple return paths						x		1
Defect Types Supported by a Tool	**all**	**6**	**10**	**7**	**5**	**14**	**1**	
	critical	4	8	4	3	7	0	
	warning	2	2	3	2	6	1	
	uncritical	0	0	0	0	1	0	

Lessons Learned

Dynamic Assumptions on Context

```
void myFunc(char *dest, const char *src) {
  dest[3]=src[0];
  dest[2]=src[1];
  dest[1]=src[2];
  dest[0]=src[3];
  return;
}
```

Tool A	Tool B	Tool A
Report	Report	Report
Report	FN ?	Report FP?
Report	FN?	Report FP?
Report	FN?	Report FP?

- Tool B reports a defect for the case: src has less than 3 elements
- alternatively, it passes dest[3] without a report: src has 3 elements at least, therefore no further reports are issued

Defect non-detectable by a tool

```
tc->elemNo_2 = buf[ind];
ind += 1;
upLim2= tc->elemNo_2;
if (upLim2>LENGTH_87)
  upLim2=LENGTH_87;
idx2=0;
while (upLim2 > idx2){
  read4Byte_86(&buf[ind], &tc>elem[idx2].elem0);
  ind += 4;
  tc->elem[idx2].elemNo_1 = buf[ind];
  ind += 1;
  upLim1= tc->elem[idx2].elemNo_1;
  if (upLim1>LENGTH_87)
     upLim1=LENGTH_87;
  idx1=0;
  while (upLim1 > idx1){

   read4Byte_86(&buf[ind],
       &tc>elem[idx2].elem_1[idx1].elem_1);
    ind += 4;
    indx1++;
  }
 }
 indx2++;
}
```

Trailing elements will not be skipped

Wrong

```
upLim1= tc->elem[idx2].elemNo_1;


idx1=0;
while (upLim1 > idx1){
  if (idx1 < LENGTH_87)
    read4Byte_86(&buf[ind],
    &tc->elem[idx2].elem_1[idx1].elem_1);
    ind += 4;
    indx1++;
  }
 }
 indx2++;
 }
```

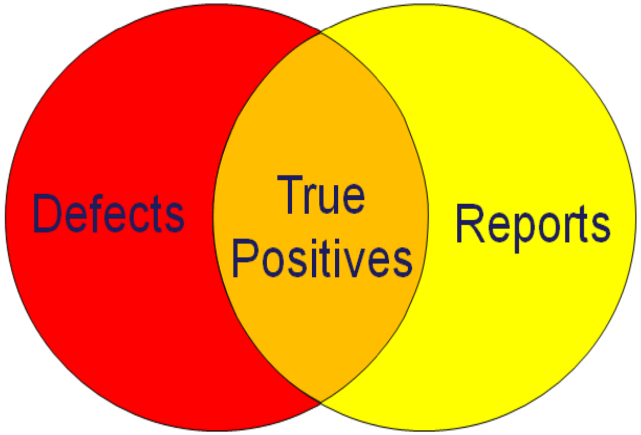
Correct

Results

Sensitivity and Precision

The higher, the more defects found

The higher, the more accurate



Sensitivity= TP/Defects

Defects = TP + FN

Precision= TP/Reports

Reports = TP + FP

- How do we approximate the true number of defects in the software?
- Approach:
  - We take the set of true positives found by all tools as an approximation
  - plus some defects found by analysis when checking the reports
  - plus some defects which never can be found by a tool (currently)

Sensitivity and Precision Summary

Tool	Criticality Level (with context, not grouped)								
	critical			warning			All (weighted)		
	TP+FP	S	P	TP+FP	S	P	TP+FP	S	P
1	9	0,04	0,78	21	0,16	1,00	30	0,09	0,93
2	138	0,40	0,51	27	0,20	1,00	165	0,32	0,59
3	101	0,44	0,78	85	0,31	0,48	186	0,39	0,65
4	55	0,30	0,98	43	0,29	0,91	98	0,30	0,95
5	100	0,44	0,78	71	0,53	1,00	171	0,48	0,78
6	0	0	n/a	2	0,02	1,00	2	0,01	1,00

Unique Contributions, Summary

Tool	Unique TP Contributions, not grouped							
	All (non-weighted)				weighted			
	ctxt		w/o ctxt		ctxt		w/o ctxt	
	TP	%	TP	%	TP	%	TP	%
1	1	0,27	1	0,26	1	0,32	1	0,31
2	29	7,86	29	7,61	29	9,32	29	9,06
3	65	17,62	56	14,70	65	20,90	56	17,50
4	27	7,32	27	7,09	27	8,68	27	8,44
5	126	34,15	131	34,38	69	22,19	71	22,19
6	2	0,54	2	0,52	2	0,64	2	0,63
Uniq	250	67,75	246	64,57	193	62,06	186	58,13
Total	369		381		311		320	

- ~ 2 / 3 of true positives are detected by one tool only
- ~ 1 / 3 of true positives are detected by more than one tool

Tool Combinations

Sensitivity if Tool B added (weighted, with context, not grouped)						
Tool A in Use	**1**	**2**	**3**	**4**	**5**	**6**
1	0,09	0,34	0,44	0,33	0,54	0,10
2	0,34	0,32	0,57	0,52	0,68	0,32
3	0,44	0,57	0,39	0,61	0,77	0,39
4	0,33	0,52	0,61	0,30	0,65	0,31
5	0,54	0,68	0,77	0,65	0,48	0,49
6	0,10	0,32	0,39	0,31	0,49	0,01

Sensitivity if Tool B added (critical, with context, not grouped)						
Tool A in Use	**1**	**2**	**3**	**4**	**5**	**6**
1	0,04	0,41	0,46	0,33	0,46	0,04
2	0,41	0,40	0,69	0,67	0,65	0,40
3	0,46	0,69	0,44	0,71	0,74	0,44
4	0,33	0,67	0,71	0,30	0,55	0,30
5	0,46	0,65	0,74	0,55	0,44	0,44
6	0,04	0,40	0,44	0,30	0,44	0,00

- one tool: up to ~ 40 – 50 %
- two tools: up to ~ 70 – 80 %

Conclusions / Outlook

About FP, TP and FN

- **False Positives**
  - position needs to be defined what the verification goal is
    - design-by-contract (with context) or
    - robustness (without context)
    - but variation is in range of 10% only

- **False Negatives**
  - found for all tools (tbc)
  - on-going discussion with one tool vendor

- **True Positives**
  - no tool did cover all defects
  - no tool **can** cover all defect types!

Sensitivity, Precision, Uniqueness

- **Sensitivity**
  - high variation depending on tool, defect type and criticality level
  - maximum 50 – 60% per tool
  - in case of combination of two tools up to 80 – 90 % can be reached
- **Precision**
  - high variation depending on tool, defect type and criticality level
  - maximum of 100% achieved by some tools and defect types
- **Uniqueness**
  - about 1/3 of defects only are found by more than one tool
  - about 2/3 of defects are found by one tool only

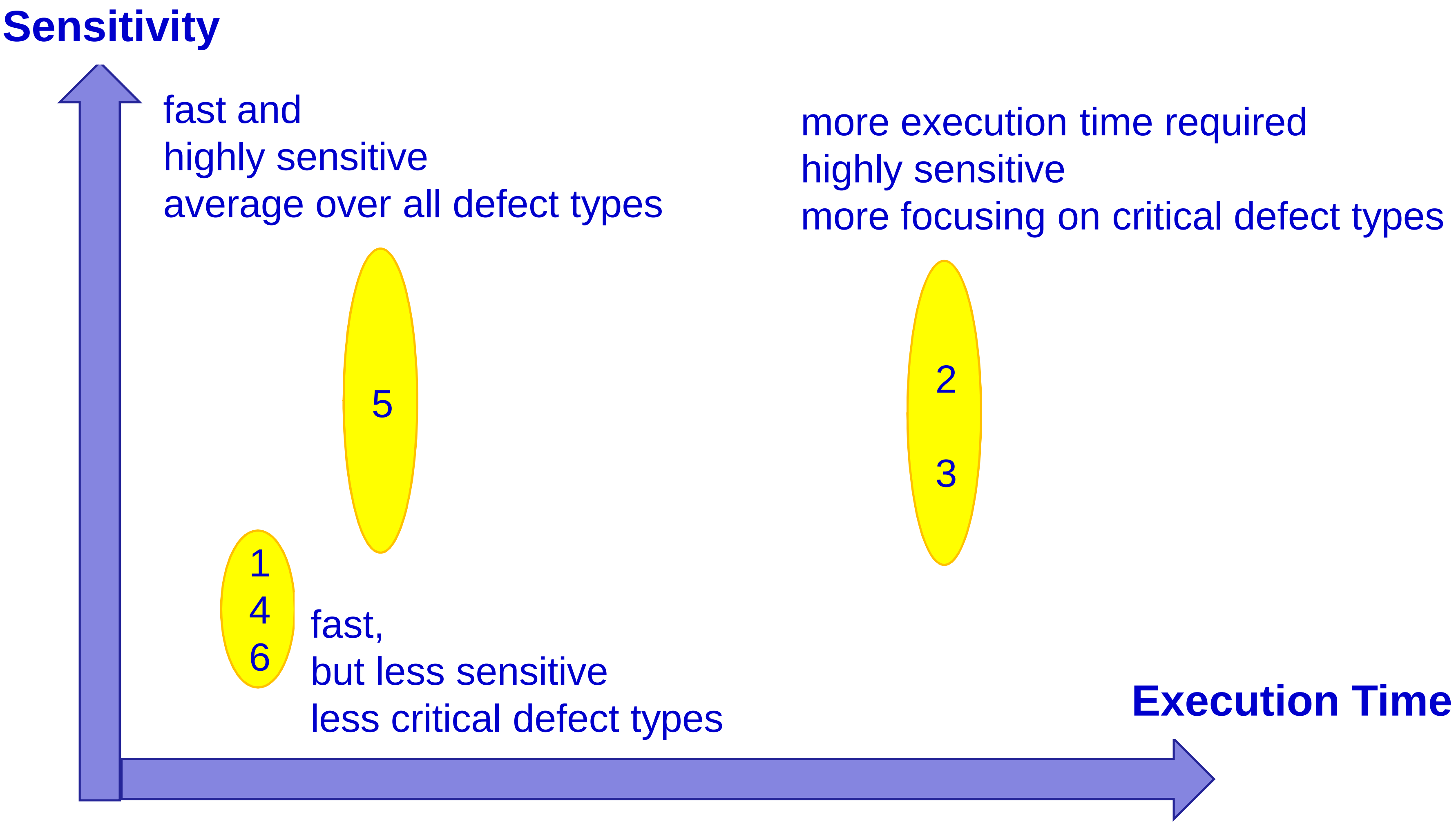
Development Process

- **Impact on Software Verification Plan and ISVV**
  - SVP: knowledge on tool profiles should be considered to minimize risks
  - ISVV: essential information on complementarity of tools
- **Programming Style**
  - early use of verification tools strongly recommended
    - Minimization of false positives
    - *Don’t complain about too many reports and FPs if best practices were not cared for right from the beginning!*
  - the engineer, not the tool drives the number of reports
    - every tool report on a (potential defect) requires effort for its classification as true or false negative in system context
    - this effort is reduced when being compliant with best practices

Future Work

- **Tool Evaluation**
  - refined and extended evaluation of recorded data
  - continuous extension of database by
    - other application software with different defect profile
    - other tools
- **Tool Support**
  - support of tool combinations
    - cost reduction by harmonized reporting
    - cost reduction by support center
  - improved reviews by hints from reports / defect distribution?
  - coverage of not reported defects by functional testing
- **Next activities**
  - possibly in 2016/2017: same software, other tools
  - in 2017: with different software and other tools (in part)

Tool Classification Sensitivity vs. Execution Time


Sensitivity
fast and
highly sensitive
average over all defect types
more execution time required
highly sensitive
more focusing on critical defect types
5
2
3
1
4
6
fast,
but less sensitive
less critical defect types
Execution Time

- Insufficient knowledge on these aspects may result in degraded performance of software verification
- This aspect should be considered in the Software Verification Plan

The project was funded by
DLR Space Administration on behalf of the
German Ministry of Economics and Energy
BMWi
under Contract No. 50 PS 1502

Thank you for your attention!

Questions?

About the Tools

Tool	1	2	3	4	5	6
			DCRTT		QA•C PRQA	gcc

Sensitivity if Tool B added (weighted, with context, not grouped)						
Tool A in Use	1	2	3	4	5	6
1	0,09	0,34	0,44	0,33	0,54	0,10
2	0,34	0,32	0,57	0,52	0,68	0,32
3	0,44	0,57	0,39	0,61	0,77	0,39
4	0,33	0,52	0,61	0,30	0,65	0,31
5	0,54	0,68	0,77	0,65	0,48	0,49
6	0,10	0,32	0,39	0,31	0,49	0,01

- disclosure of tool names as approved by tool vendor

Backup

Defect Types / Characterization

Criticality Level	Comment	Examples
Critical	The defect type does impact the correctness of system operations if being activated, i.e. it manifests to an error or a failure.	Division by zero
		NULL-ptr dererefence
		index out-of-bounds
Warning	The defect type highlights a possibly unintended operation in the source code which may, but not necessarily does manifest as a critical defect.	*Invariant condition* #define var 1 if (var==0)

		Invariant expression
		int ii=0;
		… // no modification of ii
		ii+=1;

		Unused result/ Ret-val
		ret=myFunc();
		if (var<0) printf(“ERROR in”
		“myFunc\n”);

Uncritical	The defect type is neither critical nor can it be considered a warning.	violation of a lexical or layout rule or naming convention

Evaluated Tools

Tool	Characteristics					
	Analysis		**Evaluation**			
	Type	**Analysis Approach**	**Context Sensitive**	**Context Consideration**	**Report Grouping**	**Report Export**
1	static	symbolic execution, data flow	no	n/a	no	partial
2	static	abstract interpretation	yes	auto / manual configuration	ignores follow on	full
3	dynamic	auto-stimulation	yes	auto / manual configuration	no	full
4	static	symbolic execution, dataflow	yes	auto / call tree	yes	partial
5	static	dataflow	partial	auto	no	no
6	static / compiler	syntax, type checking	no	n/a	partial	full

- Tool 6: gcc with –Wall just for comparison, no optimization

Deferred Report

```
errorCode_t enableMonitoring( const byte_t * buffer,
        const uint32_t buffer_size, enableMonitoring_t * tc ) {
    uint32_t start = 0;
    if ((buffer == NULL_POINTER) || (tc == NULL_POINTER))
       <error>
    else {
        if ( buffer_size < 1 )  // wrong check!!
           <error>
        else {
           tc->elemNo = buffer[start];
          start += 1;   // invariant expression !!
          upLim = tc->elemNo;
          if (upLim>PARA_MAX)
             upLim=PARA_MAX;
          ii=0;
          while (upLim > ii){
            read32(&buffer[start], &tc->para[ii].elem);
            start += 4;
            ii++;
        }
    }
    return ret;
}
```

```
Applied Checks                  Target    Result
Non-initialized pointer         buffer    Pointer is initialized
Non-initialized local variable  start     Local variable is initialized
Non-initialized pointer         tc        Pointer is initialized
Non-initialized local variable  ii        Local variable is initialized
```

```
Message in read32: illegally dereferenced pointer
```

- although it looks like a pointer dereferene
- pointer is not derefenced in the compiled code
- no report issued

`&buffer[start]`	original source code
`buffer+start`	expanded expression

Context Dependency

```
errorCode_t enableMonitoring( const byte_t * buffer,
        const uint32_t buffer_size, enableMonitoring_t * tc ) {
    uint32_t start = 0;
    if ((buffer == NULL_POINTER) || (tc == NULL_POINTER))
        <error>
    else {
        if ( buffer_size < 1 )  // wrong check!!
           <error>
        else {
           tc->elemNo = buffer[start];
          start += 1;   // invariant expression !!
          upLim = tc->elemNo;
          if (upLim>PARA_MAX)
             upLim=PARA_MAX;
          ii=0;
          while (upLim > ii){
            read32(&buffer[start], &tc->para[ii].elem);
            start += 4;
            ii++;
        }
    }
    return ret;
}
```

Report on an unreachable error handling branch		
Report	Context for buffer / tc	
	with NULL does not occur	without NULL may occur
issued	true positive	false positive
not issued	true negative	false negative

True Positives per Defect Type

Defect Type	Number of TP by Tool (not grouped, not weighted, Set 1)													
	1		2		3		4		5		6		All	
	with	w/o	with	w/o	with	w/o	with	w/o	with	w/o	with	w/o	with	w/o
Array Index Out-of-Bounds	2	2	33	35	60	65	51	51	47	47	0	0	118	124
Dereference of Invalid Pointer	2	2	20	31	18	30	0	0	23	32	0	0	33	47
Dereference of NULL-Pointer	0	0	1	1	0	4	0	0	3	8	0	0	3	8
File Access Error	0	0	0	0	1	1	0	0	0	0	0	0	1	1
Invalid function pointer	0	0	2	2	0	0	0	0	0	0	0	0	2	2
Non-terminating Loop	0	0	1	1	0	0	0	0	1	1	0	0	1	1
Passing invalid argument to standard library routine	0	0	1	1	0	0	0	0	0	0	0	0	1	1
(Possible) Recursion	1	1	0	0	0	0	0	0	1	1	0	0	1	1
Resource Leak	0	0	0	0	0	0	2	2	0	0	0	0	2	2
Undefined Result of Arithmetic Operation	1	1	1	1	0	0	0	0	1	1	0	0	2	2
Uninitialized Variable	2	2	12	13	0	0	1	1	2	2	0	0	14	15
Subtotal	7	7	*72*	*85*	79	100	54	54	78	92	0	0	178	204
Arithmetic Operation on NULL Pointer	0	0	0	0	1	1	0	0	1	4	0	0	1	4
Invariant Condition	0	0	1	1	11	7	3	3	6	6	0	0	16	12
Invariant Expression	0	0	0	0	0	0	0	0	44	44	0	0	44	44
Parameter Type Mismatch in Function Call	0	0	0	0	0	0	0	0	0	0	2	2	2	2
Timeout during execution	0	0	0	0	1	2	0	0	0	0	0	0	1	2
Unnecessary loop construct	0	0	0	0	0	0	0	0	1	1	0	0	1	1
Unreachable Code	17	17	26	24	29	17	36	36	7	7	0	0	56	42
Unused Result	4	4	0	0	0	0	6	6	58	58	0	0	58	58
Subtotal	21	21	*26*	*24*	42	27	39	39	117	120	2	2	179	165
Multiple return paths	0	0	0	0	0	0	0	0	12	12	0	0	12	12
Subtotal	0	0	*0*	*0*	0	0	0	0	12	12	0	0	12	12
Total	**28**	**28**	**98**	**110**	**121**	**127**	**93**	**93**	**207**	**224**	**2**	**2**	**369**	**381**
Coverage of all Defect Types in % (Sensitivity)	7,59	7,35	26,56	28,87	32,79	33,33	25,20	24,41	56,10	58,79	0,54	0,52	100,00	100,00

Tool Combinations per defect type, detailed

Defect Type	Tool 2 In use, Another Tool B Added (no context, not grouped, weighted, Set 1)																																		
	Tool 2 only							Tool 1 added							Tool 3 added							Tool 4 added							Tool 5 added						
	R	T	F	ET	EF	S	P	R	T	F	ET	EF	S	P	R	T	F	ET	EF	S	P	R	T	F	ET	EF	S	P	R	T	F	ET	EF	S	P
Array Index Out-of-Bounds	42	35	7	55	54	0,28	0,83	42	35	7	55	57	0,28	0,83	87	80	7	120	54	0,65	0,92	89	81	8	57	57	0,65	0,91	75	66	9	66	70	0,53	0,88
Dereference of Invalid Pointer	31	31	0	31	0	0,66	1,00	31	31	0	31	0	0,66	1,00	42	42	0	50	0	0,89	1,00	31	31	0	31	0	0,66	1,00	42	41	1	54	1	0,87	0,98
Dereference of NULL-Pointer	1	1	0	1	0	0,13	1,00	1	1	0	1	0	0,13	1,00	5	5	0	10	0	0,63	1,00	1	1	0	1	0	0,13	1,00	8	8	0	11	0	1,00	1,00
File Access Error	0	0	0	0	0	0,00	n/a	0	0	0	0	0	0,00	n/a	1	1	0	2	0	1,00	1,00	0	0	0	0	0	0,00	n/a	0	0	0	0	0	0,00	n/a
Invalid function pointer	2	2	0	5	0	1,00	1,00	2	2	0	5	0	1,00	1,00	2	2	0	5	0	1,00	1,00	2	2	0	5	0	1,00	1,00	2	2	0	5	0	1,00	1,00
Non-terminating Loop	1	1	0	1	0	1,00	1,00	1	1	0	1	0	1,00	1,00	1	1	0	1	0	1,00	1,00	1	1	0	1	0	1,00	1,00	1	1	0	1	0	1,00	1,00
Passing invalid argument to standard library routine	2	1	1	5	10	1,00	0,50	2	1	1	5	10	1,00	0,50	2	1	1	5	13	1,00	0,50	2	1	1	5	10	1,00	0,50	2	1	1	5	10	1,00	0,50
(Possible) Recursion	0	0	0	0	0	0,00	n/a	0	0	0	0	0	0,00	n/a	0	0	0	0	0	0,00	n/a	0	0	0	0	0	0,00	n/a	1	1	0	5	0	1,00	1,00
Resource Leak	0	0	0	0	0	0,00	n/a	0	0	0	0	0	0,00	n/a	0	0	0	0	0	0,00	n/a	2	2	0	5	0	1,00	1,00	0	0	0	0	0	0,00	n/a
Undefined Result of Arithmetic Operation	1	1	0	2	0	0,50	1,00	2	2	0	4	0	1,00	1,00	1	1	0	2	0	0,50	1,00	1	1	0	2	0	0,50	1,00	2	2	0	4	0	1,00	1,00
Uninitialized Variable	58	13	45	11	9	0,87	0,22	59	14	45	13	9	0,93	0,24	58	13	45	11	9	0,87	0,22	58	13	45	12	9	0,87	0,22	59	14	45	14	9	0,93	0,24
Arithmetic Operation on NULL Pointer	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a
Invariant Condition	1	1	0	0	0	0,08	1,00	1	1	0	0	0	0,08	1,00	13	7	6	5	6	0,58	0,54	5	3	2	5	7	0,25	0,60	7	7	0	13	0	0,58	1,00
Invariant Expression	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a
Parameter Type Mismatch in Function Call	0	0	0	0	0	0,00	n/a	0	0	0	0	0	0,00	n/a	0	0	0	0	0	0,00	n/a	0	0	0	0	0	0,00	n/a	0	0	0	0	0	0,00	n/a
Timeout during execution	0	0	0	0	0	0,00	n/a	0	0	0	0	0	0,00	n/a	2	2	0	21	0	1,00	1,00	0	0	0	0	0	0,00	n/a	0	0	0	0	0	0,00	n/a
Unnecessary loop construct	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a
Unreachable Code	26	24	2	29	2	0,57	0,92	27	25	2	32	2	0,60	0,93	84	29	55	47	175	0,69	0,35	41	37	4	39	4	0,88	0,90	29	27	2	36	2	0,64	0,93
Unused Result	0	0	0	0	0	0,00	n/a	4	4	0	4	0	0,07	1,00	0	0	0	0	0	0,00	n/a	0	0	0	0	0	0,00	n/a	58	58	0	64	0	1,00	1,00
Multiple return paths	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a	0	0	0	0	0	n/a	n/a
All	165	110	55	140	75	0,34	0,67	172	117	55	152	78	0,37	0,68	298	184	114	278	257	0,58	0,62	233	173	60	163	87	0,54	0,74	286	228	58	278	92	0,71	0,80