LinuxTesting.org

The Experience of Heavy Weight Static Analysis of Linux Device Drivers

Alexey Khoroshilov
khoroshilov@linuxtesting.org

ISP RAS
Institute for System Programming of the Russian Academy of Sciences

Outline

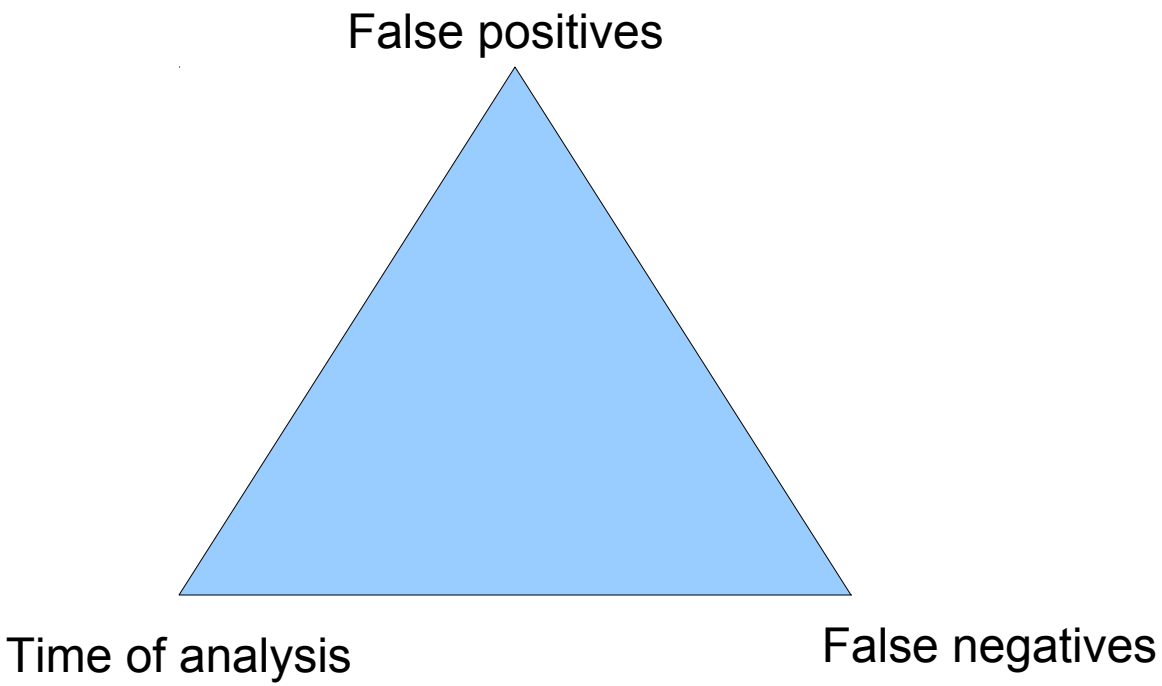
- Heavy Weight Static Analysis
- Linux Driver Verification
- Lessons Learnt

Static Analysis

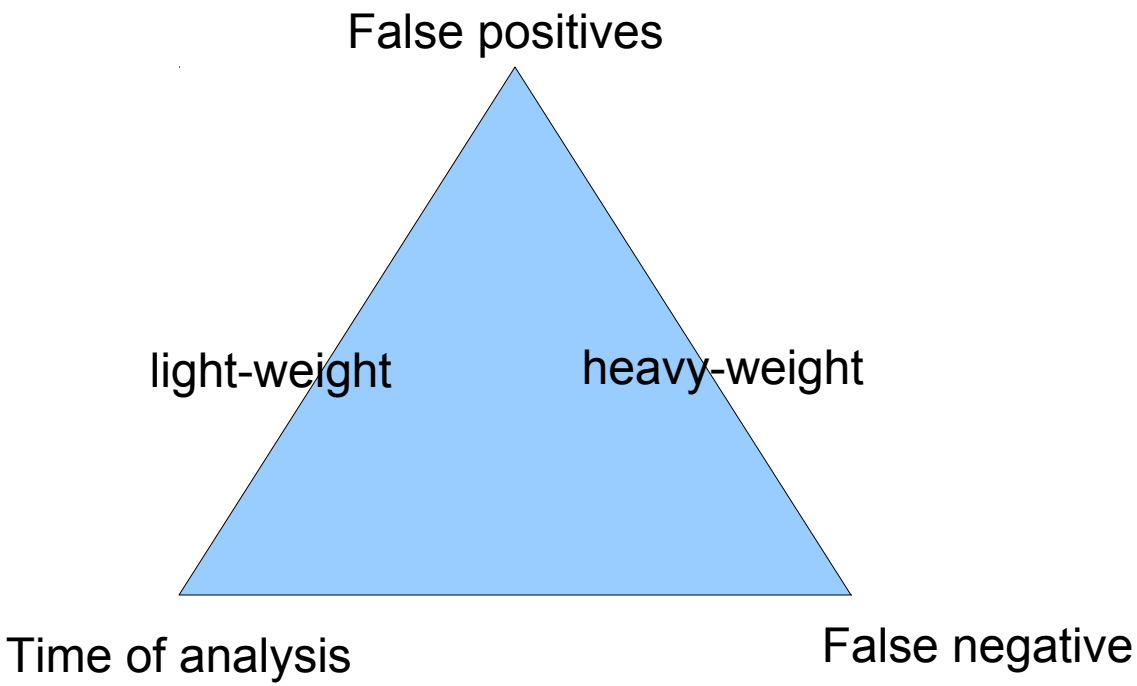
Key characteristics

- Scope of analysis (kind of bugs)
- False positives (false bugs reported)
- False negatives (real bugs missed)
- Resources required for analysis

Static Analysis: Trade-Off Triangle



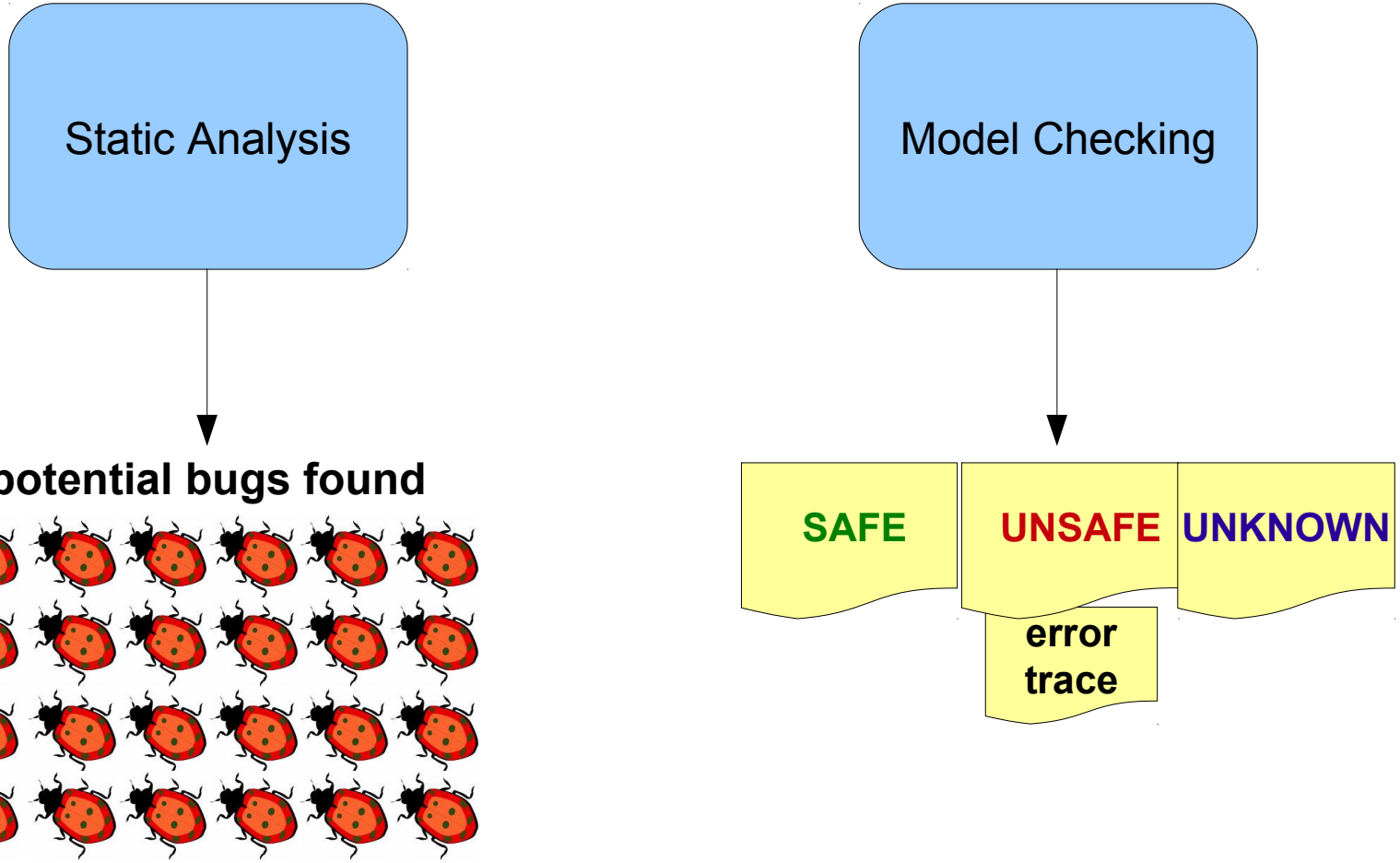
Static Analysis: Trade-Off Triangle



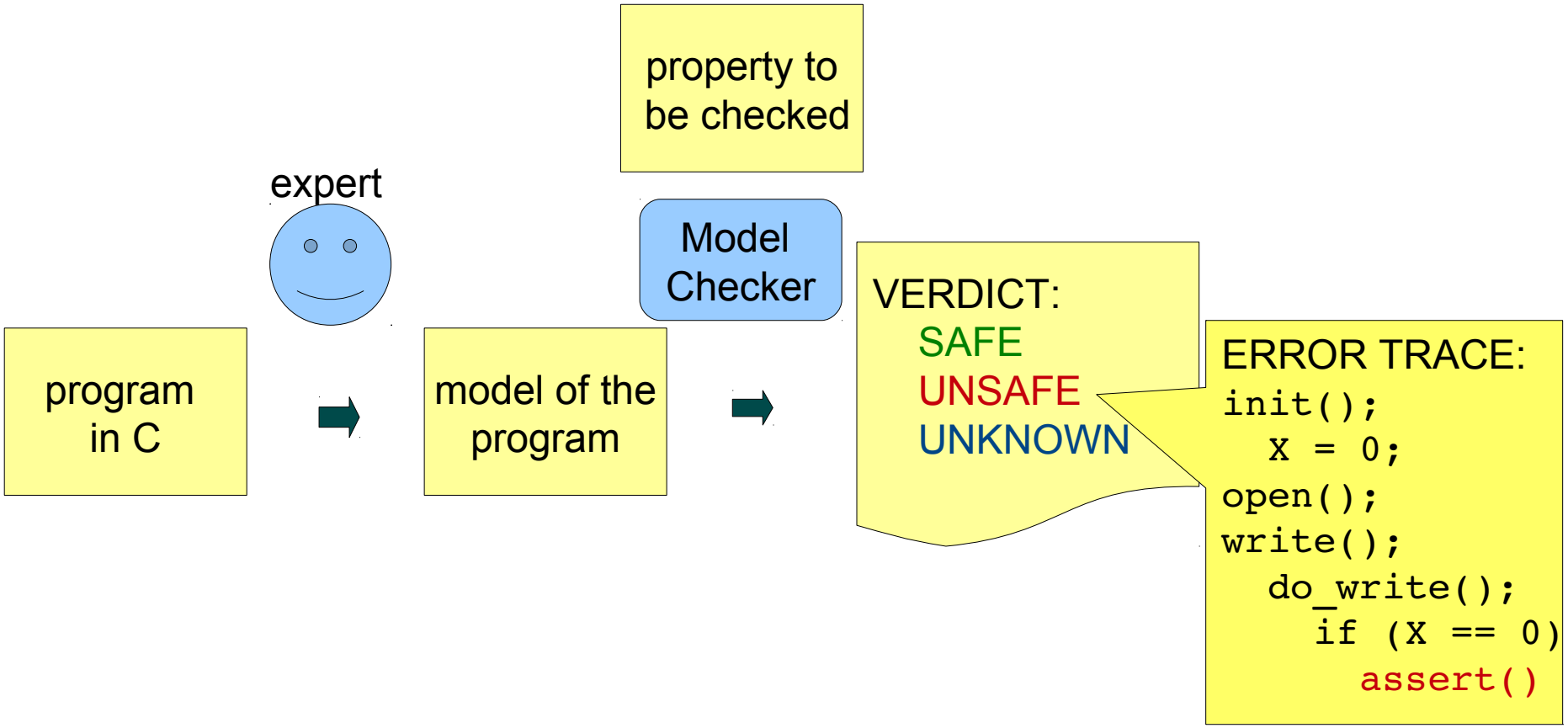
Heavy-Weight Analysis

Based on a picture from http://engineer.org.in

Static Analysis vs Model Checking

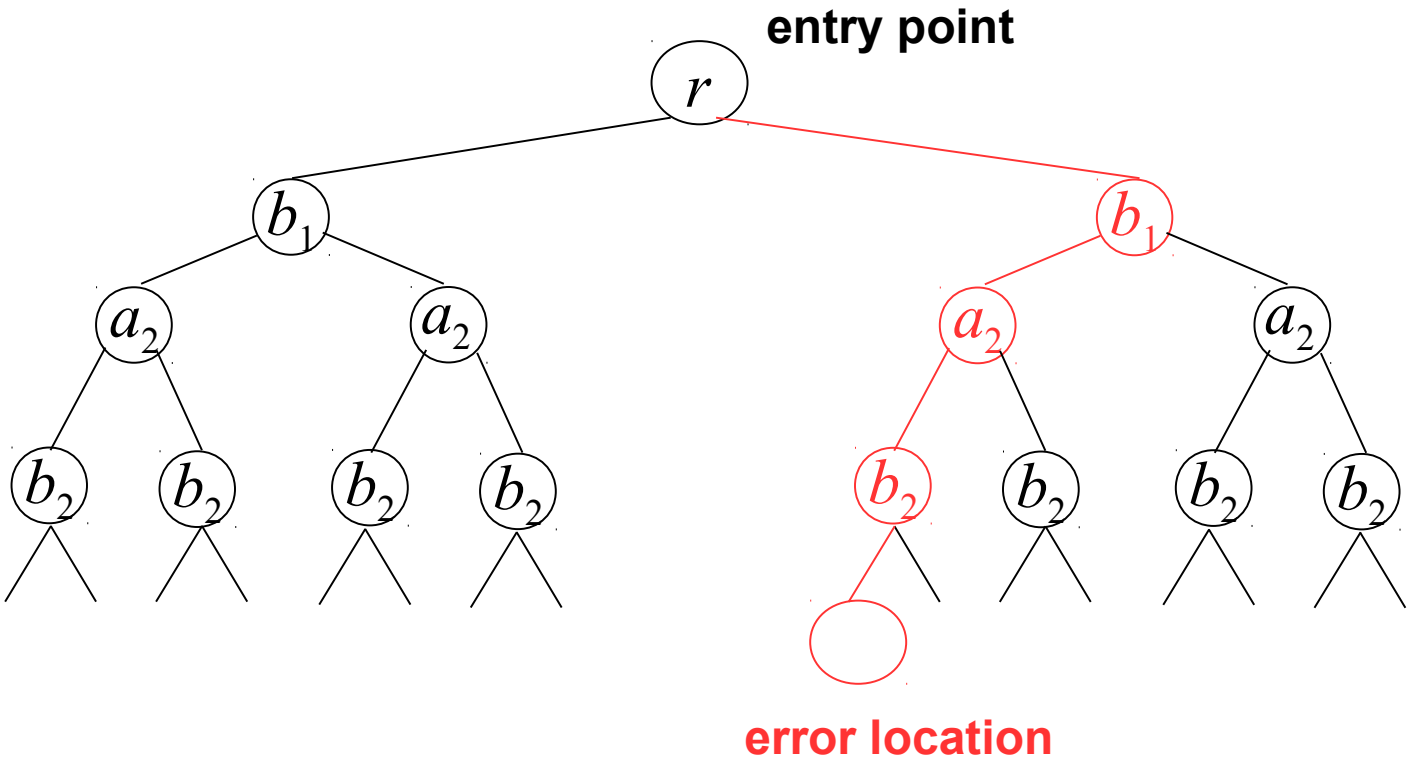


Model Checking: Originally



Model Checking: Inside

- Reachability problem

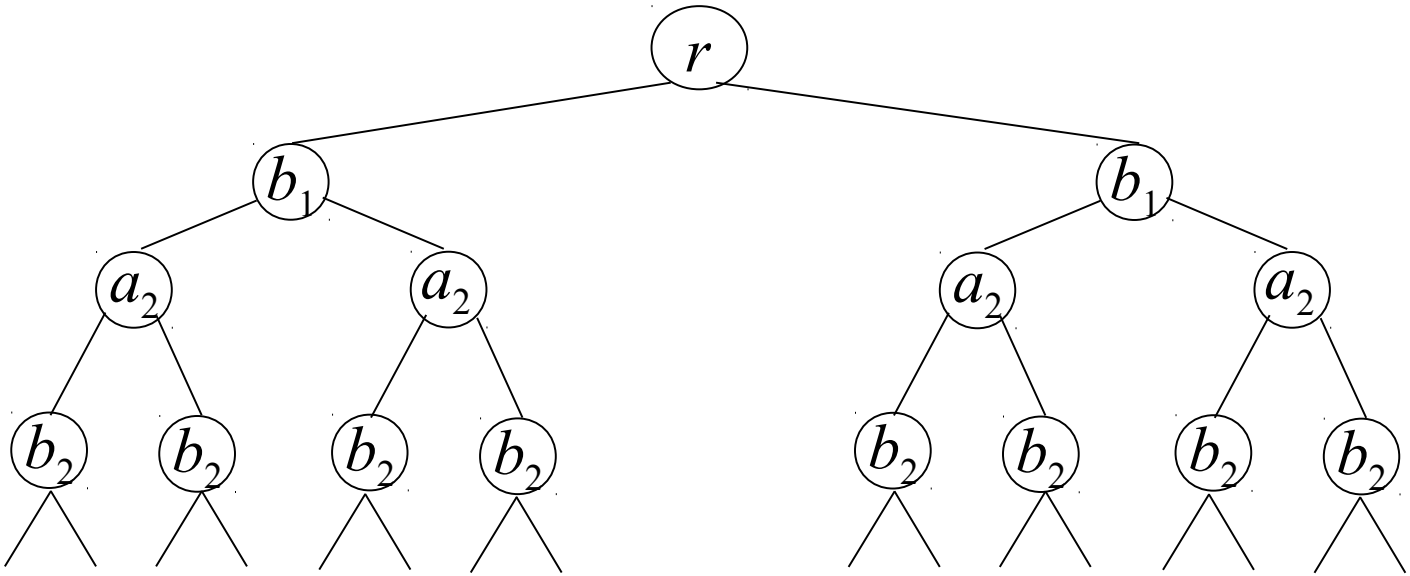


Model Checking: Now

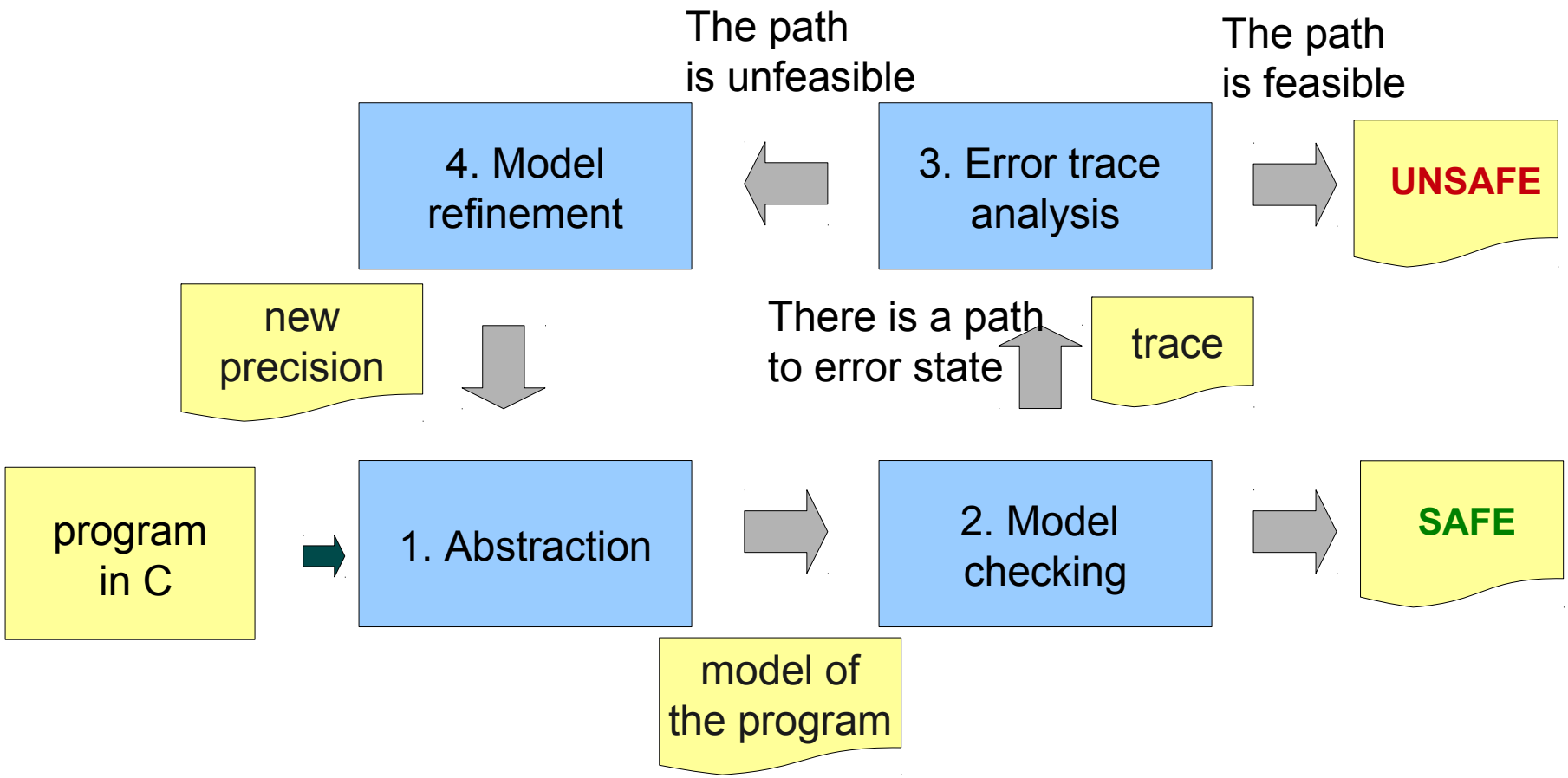
- BMC – **B**ounded **M**odel **C**hecking
- CEGAR – **C**ounter-**E**xample **G**uided **A**bstraction **R**efinement

Bounded Model Checking

- finite unfolding of transition relation



Counter-Example Guided Abstraction Refinement



ETAPS
EUROPEAN JOINT CONFERENCES ON THEORY & PRACTICE OF SOFTWARE

TACAS 2012

Competition on Software Verification (SV-COMP)

We mourn the passing of competition participant Daniel Wonisch. (Feb. 21, 2012)

TACAS'12
March 2012
Tallinn, Estonia

1st Intl. Competition on Software Verification held at TACAS 2012 in Tallinn, Estonia.

Motivation

Competition is a driving force for the invention of new methods, technologies, and tools. This web page describes the competition of software-verification tools, which will take place at TACAS'12.

There are several new and powerful software-verification tools around, but they are very difficult to compare. The reason is that no widely distributed benchmark suite is available and most concepts are only validated in research prototypes. This competition wants to change this.

Only few projects aim at producing stable tools that can be used by people outside the respective development groups, and the development of such tools is not continuous. Also, PhD students and PostDocs do not adequately benefit from tool development because theoretical papers count more

http://sv-comp.sosy-lab.org

SVCOMP'12 Results

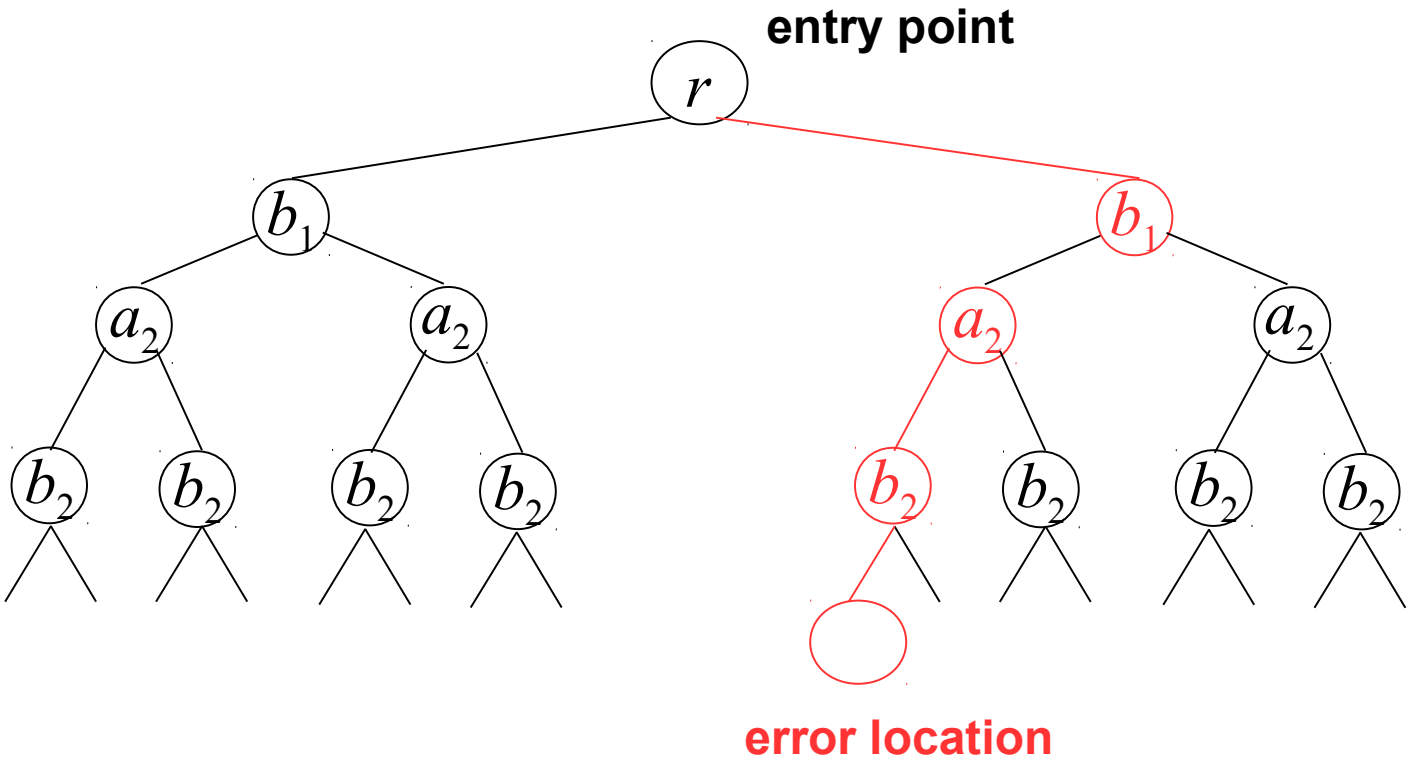
	CEGAR	CEGAR	CEGAR	BMC	BMC	BMC	CEGAR	CEGAR	CEGAR
Competition candidate	**BLAST 2.7**	**CPAchecker ABE 1.0.10**	**CPAchecker Memo 1.0.10**	**ESBMC 1.17**	**FShell 1.3**	**LLBMC 0.9**	**Predator 20111011**	**QARMC -HSF**	**SATabs 3.0**	**Wolverine 0.5c**
Affiliation	Moscow, Russia	Passau, Germany	Paderborn, Germany	Southampton, UK	Vienna, Austria	Karlsruhe, Germany	Brno, Czechia	Munich, Germany	Oxford, UK	Princeton, USA
ControlFlowInteger 93 files, max score: 144	71 9900 s	141 1000 s	140 3200 s	102 4500 s	28 580 s	100 2400 s	17 1100 s	140 4800 s	75 5400 s	39 580 s
DeviceDrivers 59 files, max score: 103	72 30 s	51 97 s	51 93 s	63 160 s	20 3.5 s	80 1.6 s	80 1.9 s	--	71 140 s	68 65 s
DeviceDrivers64 41 files, max score: 66	55 1400 s	26 1900 s	49 500 s	10 870 s	0 0 s	1 110 s	0 0 s	--	32 3200 s	16 1300 s
HeapManipulation 14 files, max score: 24	--	4 16 s	4 16 s	1 220 s	--	17 210 s	20 1.0 s	--	--	--
SystemC 62 files, max score: 87	33 4000 s	45 1100 s	36 450 s	67 760 s	--	8 2.4 s	21 630 s	8 820 s	57 5000 s	36 1900 s
Concurrency 8 files, max score: 11	--	0 0 s	0 0 s	6 270 s	0 0 s	--	0 0 s	--	1 1.4 s	--
Overall 277 files, max score: 435	231 15000 s	267 4100 s	280 4300 s	249 6800 s	48 580 s	206 2700 s	138 1700 s	148 5600 s	236 14000 s	159 3800 s

Outline

- Heavy Weight Static Analysis
- **Linux Driver Verification**
- Lessons Learnt

Model Checking and Linux Kernel

- Reachability problem



Verification Tools World

```
int main(int argc,char* argv[])
{
 ...
 other_func(var);
 ...
}
```

```
void other_func(int v)
{
   ...
   assert( x != NULL);
}
```

Device Driver World

```
static struct pci_driver DAC960_pci_driver = {
        .name           = "DAC960",
        .id_table       = DAC960_id_table,
        .probe          = DAC960_Probe,
        .remove         = DAC960_Remove,
};

static int DAC960_init_module(void)
{
        int ret;
        ret = pci_register_driver(&DAC960_pci_driver)
#ifdef DAC960_GAM_MINOR
        if (!ret)
                DAC960_gam_init();
#endif
        return ret;
}
...
module_init(DAC960_init_module);
module_exit(DAC960_cleanup_module);
```

Callback interface procedures registration

No explicit calls to linking-level init procedures

Pseudo-main generation

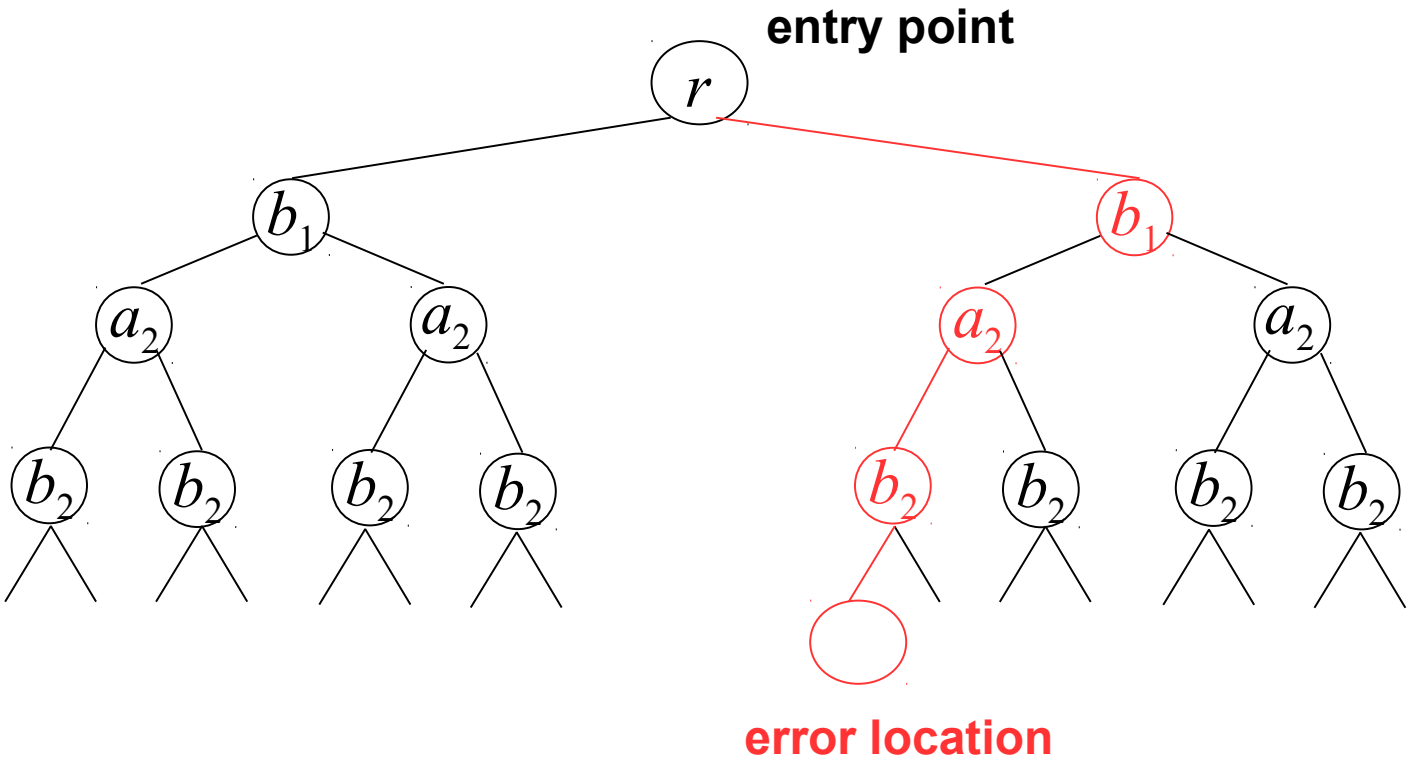
```
int main(int argc,char* argv[])
{
  init_module()
  for(;;) {
    switch(*) {
     case 0: driver_probe(*,*,*);break;
     case 1: driver_open(*,*);break;
     ...
    }
  }
  exit_module();
}
```

Pseudo-main generation (2)

- Order limitation
  - open() after probe(), but before remove()
- Implicit limitations
  - read() only if open() succeed
- and it is specific for each class of drivers

Model Checking and Linux Kernel

- Reachability problem



Rule Instrumentor

```
mutex x;
int f(int y)
{
   lock(x);
   ...
   unlock(x);
   return y;
}
```

```
int x_locked = 0;
int f(int y)
{
   assert(x_locked == 0);
   x_locked = 1;
   ...
   assert(x_locked == 1);
   x_locked = 0;
   return y;
}
```

Aspect-Oriented Approach

```
mutex x;
int f(int y)
{
   lock(x);
   ...
   unlock(x);
   return y;
}
```

Aspect:

```
around:
call(int lock(mutex x)
{
 assert(x_locked == 0);
 x_locked = 1;
}
```

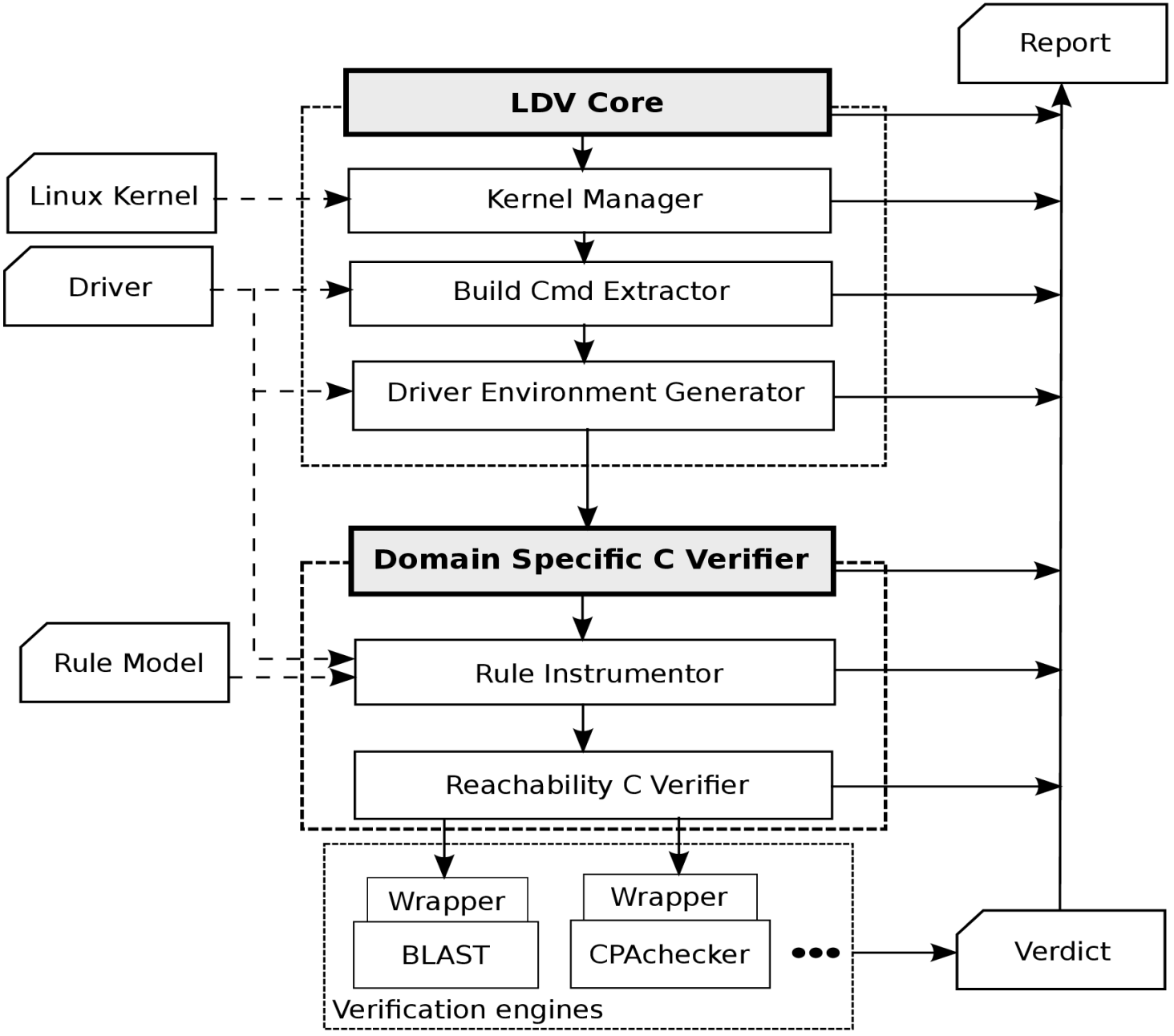
Rule Instrumentor

```
mutex x;
int f(int y)
{
   lock(x);
   ...
   unlock(x);
   return y;
}
```

```
int x_locked = 0;
int f(int y)
{
   assert(x_locked == 0);
   x_locked = 1;
   ...
   assert(x_locked == 1);
   x_locked = 0;
   return y;
}
```

Rule Instrumentor: Implementation

- **CIF** – C Instrumentation Framework
  - gcc-based aspect-oriented programming tool for C language
  - available at forge.ispras.ru under GPLv3


Report
LDV Core
Linux Kernel
Kernel Manager
Driver
Build Cmd Extractor
Driver Environment Generator
Domain Specific C Verifier
Rule Model
Rule Instrumentor
Reachability C Verifier
Wrapper
BLAST
Wrapper
CPAchecker
Verdict
Verification engines

Where we are

- Static analysis infrastructure
- Front-ends
  - ldv-manager
  - ldv-git
  - ldv-online

ldv-online

Online Linux Driver Verification Service (alpha)

Start Verification | Verification History | Rules

Start Verification

on x86_64 architecture

1. Ensure that drivers satisfy the following requirements:

- The driver is archived using gzip or bzip2 and has one of the following extensions: .tar.bz2, tar.gz, .tgz
- Archive should contain:
  - o Makefile (written to be compiled with the kernel)
  - + obj-m is mandatory
  - o Sources needed by Makefile
- Archive should not contain generated files left from builds

2. Upload driver.

3. Wait for results.

Browse...

Start Verification

ldv-online (2)

Verification Report

Driver: test-0032-wl12xx-unsafe.tar.bz2
Timestamp: 2011-01-19 20:51:12
Verification architecture: x86_64

You can see **verification verdict** for each rule and linux kernel. Verdict may be:

- ***Safe*** - there is no mistakes for the given linux kernel and rule.
- ***Unsafe*** - driver may contain an error. You can see the error trace by clicking on the "Unsafe" link for the corresponding linux kernel and rule.
- ***Build failed*** - your driver is not compatible with the given linux kernel. In this case you may see the compile error trace by clicking on the "more details" link.
- ***Unknown*** - tools can not determine whether your driver *Safe* or *Unsafe*.
- ***Queued*** - the driver waits for the turn to verification.

7%

linux-2.6.32.12	
Rule	**Verdict**
Mutex lock/unlock	Unsafe
NOIO allocation under usb_lock	Safe
Module get/put	
PCI pool create/destroy, alloc/free	Queued
Delay in probe_irq on/off	Queued
Memory allocation inside spinlocks	Queued
Linked list double add	Queued
Usb alloc/free urb	Queued
Spinlocks lock/unlock	Queued

Where we are

- Static analysis infrastructure
- Cluster framework
- Front-ends
  - ldv-manager
  - ldv-git
  - ldv-online
- Results database
  - Error trace visualizer
  - Knowledge base
  - Comparison framework

Error Trace Visualizer

Rule: Mutex lock/unlock

Error trace

☑ Function bodies | ☑ Blocks | Others...

```
3182    LDV_IN_INTERRUPT = 1;
3191    +ldv_initialize_FOREACH();
3195    tmp___8 = nondet_int() { /* The function body
3195    assert(tmp___8 != 0);
3198    tmp___7 = nondet_int() { /* The function body
3200    assert(tmp___7 != 0);
3280    assert(tmp___7 != 1);
3360    assert(tmp___7 != 2);
3440    assert(tmp___7 != 3);
3520    assert(tmp___7 != 4);
3600    assert(tmp___7 != 5);
3680    assert(tmp___7 != 6);
3760    assert(tmp___7 != 7);
3840    assert(tmp___7 != 8);
3920    assert(tmp___7 != 9);
4000    assert(tmp___7 != 10);
4080    assert(tmp___7 == 11);
4130    -carl9170_op_set_key(var_group1 /* hw */
        {
1031        -ar = *(hw).priv;
            err = 0;
1035        assert(*(ar).disable_offload == 0);
1035        assert(vif != 0);
1047        +tmp___7 = is_main_vif(ar /* ar */, v
1047        assert(tmp___7 == 0);
1159        assert(*(ar).rx_software_decryption
1163        +mutex_unlock_mutex(&(ar)->mutex /*
        }
    }
```

Source code

carl9170.h | **main.c.common.c** | wlan.h | rcupdate.h

```
1026 static int carl9170_op_set_key(struct ieee80211_hw *hw, enum set_key_cr
1027                              struct ieee80211_vif *vif,
1028                              struct ieee80211_sta *sta,
1029                              struct ieee80211_key_conf *key)
1030 {
1031   struct ar9170 *ar = hw->priv;
1032   int err = 0, i;
1033   u8 ktype;
1034
1035   if (ar->disable_offload || !vif)
1036           return -EOPNOTSUPP;
1037
1038   /*
1039    * We have to fall back to software encryption, whenever
1040    * the user choose to participates in an IBSS or is connected
1041    * to more than one network.
1042    *
1043    * This is very unfortunate, because some machines cannot handle
1044    * the high througput speed in 802.11n networks.
1045    */
1046
1047   if (!is_main_vif(ar, vif))
1048           goto err_softw;
1049
1050   /*
1051    * While the hardware supports *catch-all* key, for offloading
1052    * group-key en-/de-cryption. The way of how the hardware
1053    * decides which keyId maps to which key, remains a mystery...
1054    */
```

Knowledge Base

#	Task	Kernel	Rule	Total	Safe	Unsafe	Unknown	Verdicts				
								True	False	?	↯	
1	Task description **May, 2011**	linux-2.6.38.2	Fail before RI	75	-	-	75	-	-	-	-	
2			32_1a	2747	2077	66	604	-	-	-	-	
3			32_7	2747	2227	20	500	3	13	-	-	
4			39_7	2747	2244	15	488	2	11	-	-	
5			68_1	2747	2129	68	550	2	26	-	-	
6		linux-2.6.39	Fail before RI	81	-	-	81	-	-	-	-	
7			32_7	2826	2278	21	527	2	19	-	-	
8	Task description **June 2011**	linux-2.6.39	Fail before RI	81	-	-	81	-	-	-	-	
9			08_1	2826	2124	50	652	-	-	-	-	
10			32_7	2826	2292	29	505	5	24	-	-	
11			39_7	2826	2319	19	488	4	15	-	-	
12			43_1a	2826	1861	7	958	1	5	-	-	
13			68_1	2826	2186	76	564	2	28	-	-	
14	Task description **August 2011**	linux-3.0.1	Fail before RI	102	-	-	102	-	-	-	-	
15			08_1	3203	2550	66	587	-	-	1	-	
16			32_7	3203	2631	43	529	11	32	-	-	
17			39_7	3203	2659	24	520	5	19	-	-	
18			43_1a	3203	2623	8	572	1	7	-	-	
19			68_1	3203	2524	90	589	3	58	1	-	

Bugs Found

http://linuxtesting.org/results/ldv

- 50 patches already applied

Problems in Linux Kernel

This section contains information about problems in Linux kernel found within Linux Driver Verification program.

No.	Type	Brief	Added on	Accepted	Status
L0050	Crash	carl9170: unlock of unheld mutex in carl9170_op_set_key	2011-08-30	https://lkml.org/lkml/2011/8/23/380 commit	Fixed in kernel 3.1-rc5
K0009	Leak	(ath5k) sc->ah is allocated in ath5k_init_softc() but is not freed	2011-08-08	Kernel Bug Tracker, bug #37592	Fixed in the kernel 3.1-rc1
L0049	Crash	hfsplus: Fix double iput of the same inode in hfsplus_fill_super()	2011-06-24	https://lkml.org/lkml/2011/6/23/675 commit	Fixed in kernel 3.0
L0048	Crash	hfsplus: add error checking for hfs_find_init()	2011-06-24	https://lkml.org/lkml/2011/7/5/500 commit	Fixed in kernel 3.1-rc1
L0047	Leak	drivers/video/hecubafb.c: absence of module_put on an error path in hecubafb_probe()	2011-06-20	https://lkml.org/lkml/2011/6/17/267 commit	Fixed in kernel 3.0-rc6
L0046	Leak	gigaset: absence of call module_put before restart of if_open()	2011-06-20	https://lkml.org/lkml/2011/6/17/321 commit 2f9381e	Fixed in kernel 3.0-rc4
L0045	Leak	drivers/net/wan/farsync.c: module_get() without module_put()	2011-06-20	https://lkml.org/lkml/2011/6/17/320 commit d0fd64c	Fixed in kernel

Outline

- Heavy Weight Static Analysis
- Linux Driver Verification
- **Lessons Learnt**

Lessons Learnt

- Language features support

No matters which advanced techniques implemented by a tool if it does not work on your code

Lessons Learnt

- Language features support
- Efficiently ignore irrelevant details

Ten of thousands irrelevant transitions vs. dozens of relevant ones

Lessons Learnt

- Language features support
- Efficiently ignore irrelevant details
- No premature UNKNOWN

```
Error: Unsupported C feature (recursion) in line 60858:
    tmp = gma_power_begin( tmp24, tmp25);
    (CallstackTransferRelation.getAbstractSuccessors)
```

Bug Finder vs. Safe Prover

Lessons Learnt

- Language features support
- Efficiently ignore irrelevant details
- No premature UNKNOWN
- Pointer analysis is a weak point

  complex data structures

  containerof

  even arrays

  many false positives for complex rules

Lessons Learnt

- Language features support
- Efficiently ignore irrelevant details
- No premature UNKNOWN
- Pointer analysis is a weak point
- Engineering Matters

BLAST

Berkeley

Lazy

Abstraction

Software Verification

Tool

BLAST is a software model checker for C programs.

It uses counterexample-driven automatic abstraction refinement to construct an abstract model which is model checked for safety properties.

ISPRAS BLAST 2.6 Release Notes

Speedup ranges from **8 times** on small-sized programs to **30 times** on medium-sized programs

- Logarithmic algorithm for useful-blocks (significantly speedup of trace analysis)
- Improved integration with SMT solvers
  - efficient string concatenation
  - caching of converted formulae
  - optimization of CVC3 options for BLAST use cases
- Formulae normalization moved to solvers since solvers do it faster
- Alias analysis speedup
  - must-aliases are handled separately and faster than may-aliases
  - removed unnecessary debug prints from alias iteration (even a check for debug flag impacts performance significantly in hot places)
- BLAST-specific tuning of OCaml virtual machine options

SVCOMP'12 Results

	CEGAR	CEGAR	CEGAR	BMC	BMC	BMC	CEGAR	CEGAR	CEGAR
Competition candidate	**BLAST 2.7**	**CPAchecker ABE 1.0.10**	**CPAchecker Memo 1.0.10**	**ESBMC 1.17**	**FShell 1.3**	**LLBMC 0.9**	**Predator 20111011**	**QARMC -HSF**	**SATabs 3.0**	**Wolverine 0.5c**
Affiliation	Moscow, Russia	Passau, Germany	Paderborn, Germany	Southampton, UK	Vienna, Austria	Karlsruhe, Germany	Brno, Czechia	Munich, Germany	Oxford, UK	Princeton, USA
ControlFlowInteger 93 files, max score: 144	71 9900 s	141 1000 s	140 3200 s	102 4500 s	28 580 s	100 2400 s	17 1100 s	140 4800 s	75 5400 s	39 580 s
DeviceDrivers 59 files, max score: 103	72 30 s	51 97 s	51 93 s	63 160 s	20 3.5 s	80 1.6 s	80 1.9 s	--	71 140 s	68 65 s
DeviceDrivers64 41 files, max score: 66	55 1400 s	26 1900 s	49 500 s	10 870 s	0 0 s	1 110 s	0 0 s	--	32 3200 s	16 1300 s
HeapManipulation 14 files, max score: 24	--	4 16 s	4 16 s	1 220 s	--	17 210 s	20 1.0 s	--	--	--
SystemC 62 files, max score: 87	33 4000 s	45 1100 s	36 450 s	67 760 s	--	8 2.4 s	21 630 s	8 820 s	57 5000 s	36 1900 s
Concurrency 8 files, max score: 11	--	0 0 s	0 0 s	6 270 s	0 0 s	--	0 0 s	--	1 1.4 s	--
Overall 277 files, max score: 435	231 15000 s	267 4100 s	280 4300 s	249 6800 s	48 580 s	206 2700 s	138 1700 s	148 5600 s	236 14000 s	159 3800 s

Conclusions

- Language features support
- Efficiently ignore irrelevant details
- No premature UNKNOWN
- Pointer analysis is a weak point
- Engineering Matters

LinuxTesting.org

Thank you!

Alexey Khoroshilov
khoroshilov@linuxtesting.org
http://linuxtesting.org/project/ldv

ISP RAS
Institute for System Programming of the Russian Academy of Sciences