LinuxTesting.org

Linux Driver Verification Program

Alexey Khoroshilov
khoroshilov@linuxtesting.org

ISP RAS

Institute for System Programming of the Russian Academy of Sciences

Yet another static analysis tool?

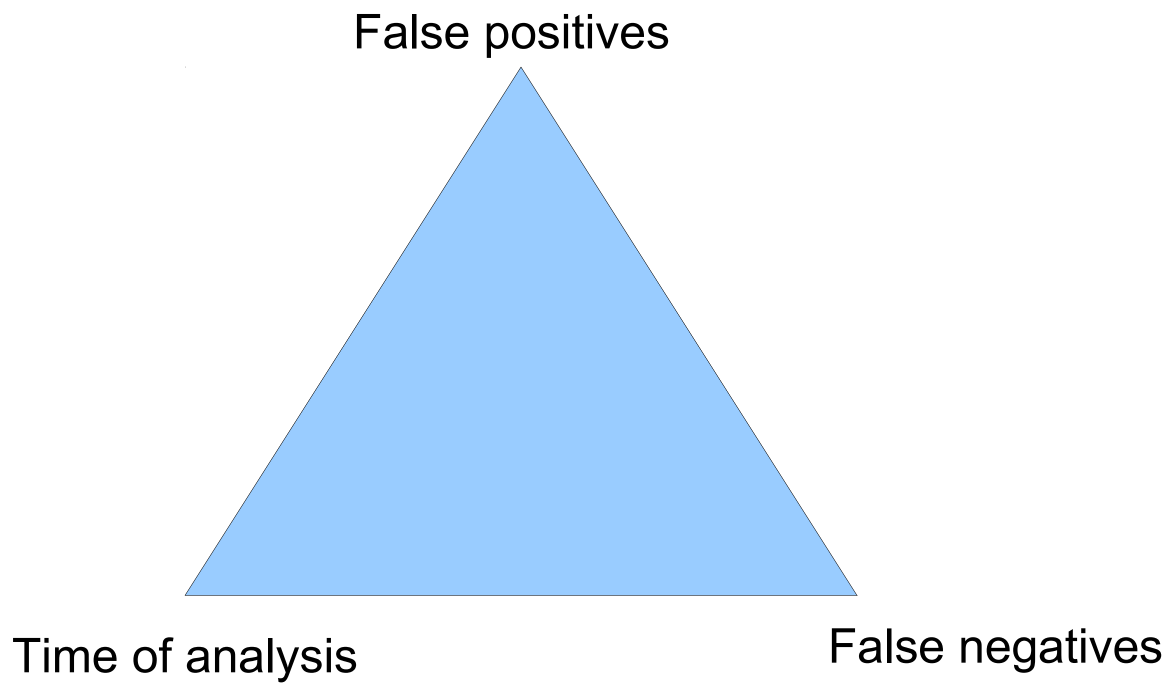
- sparse
- Coccinelle

Static Analysis

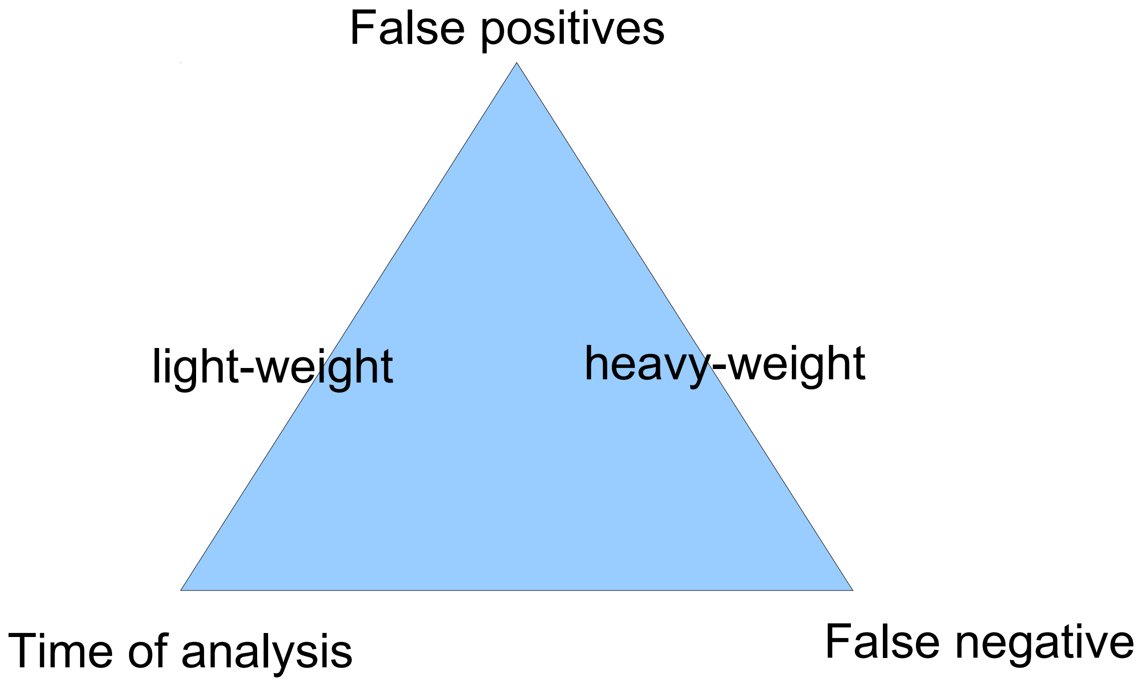
Key characteristics

- 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



Coccinelle

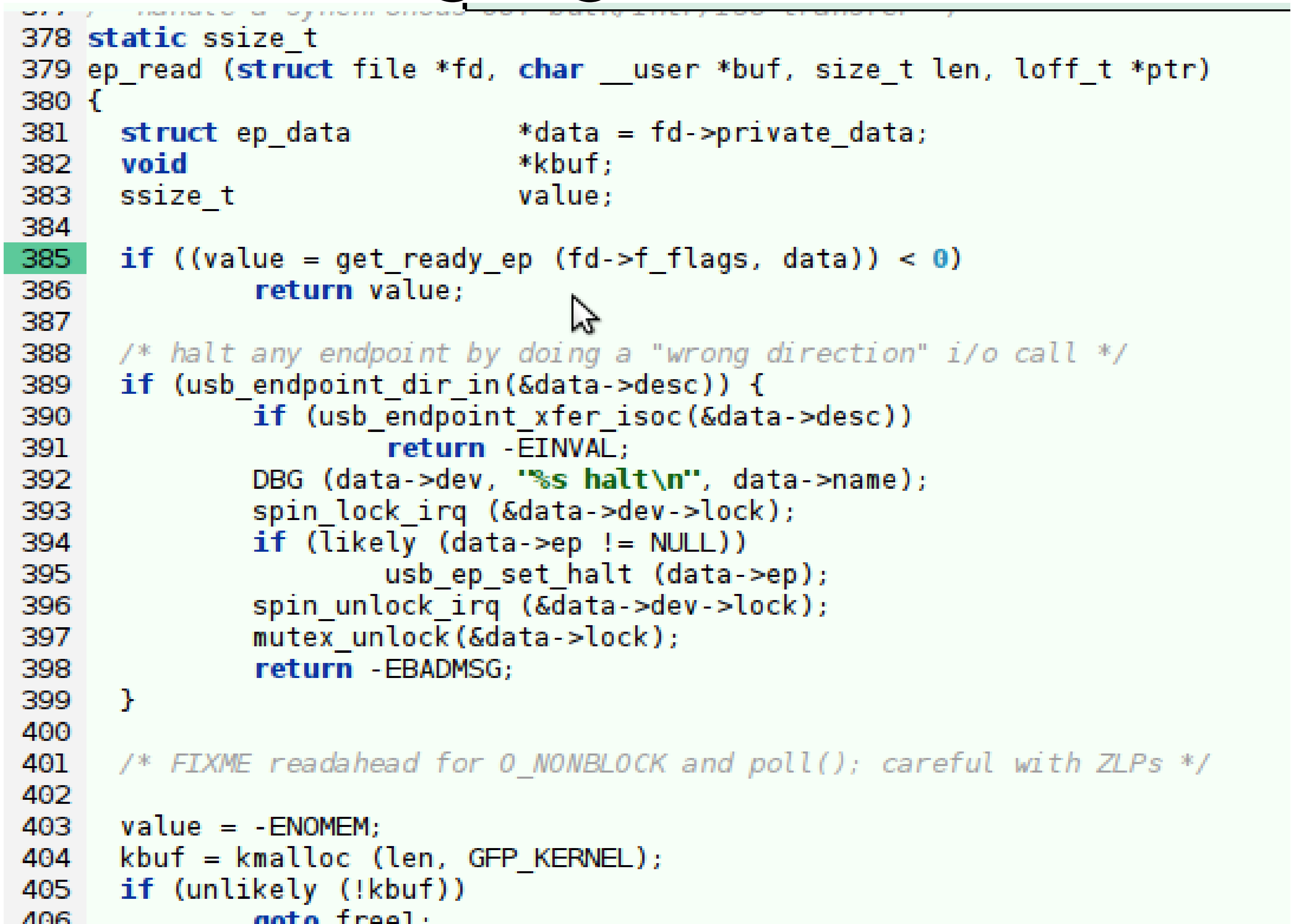
- Intra-procedural analysis
- Limited data-flow analysis

The simplest rule

mutex

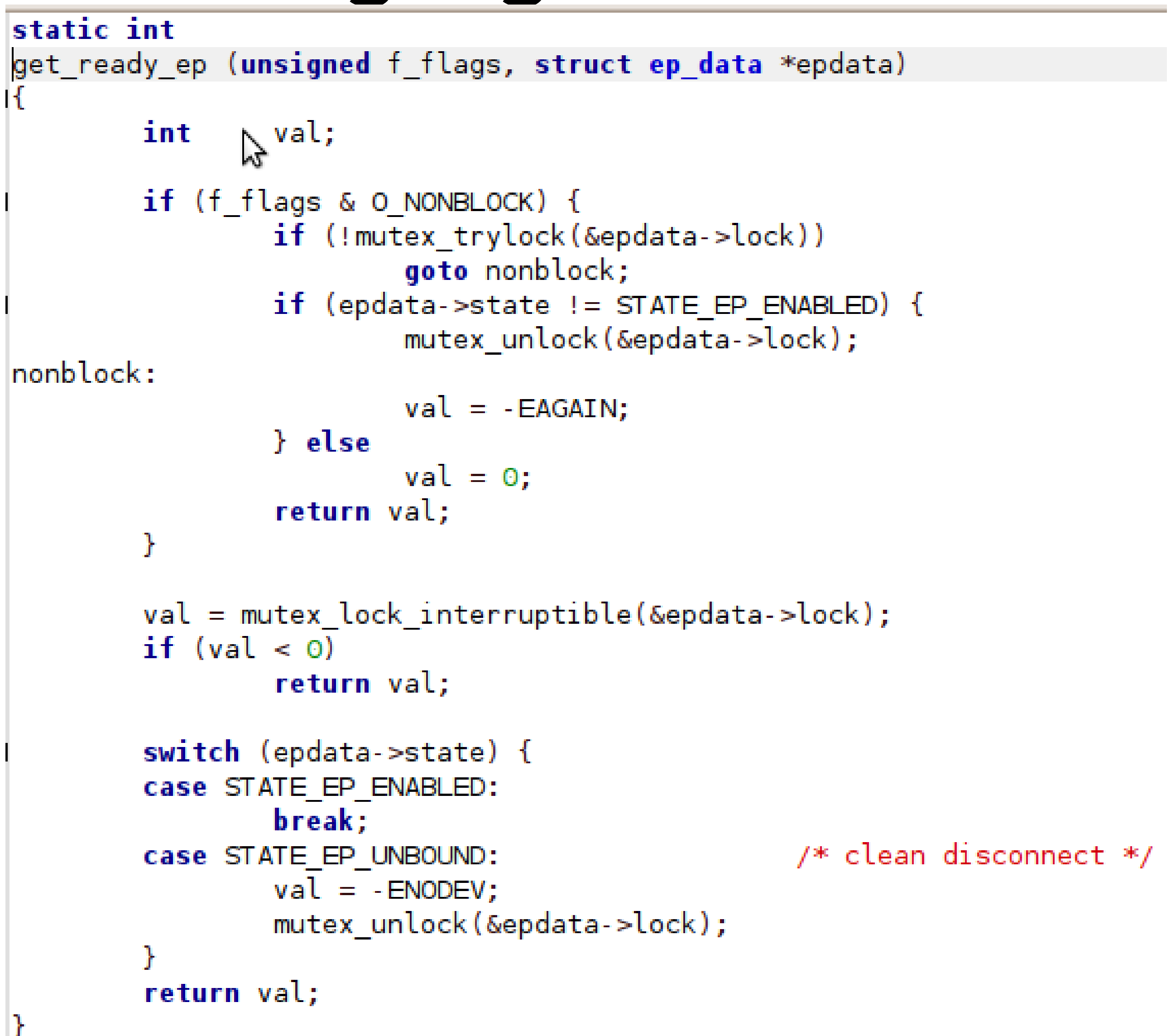
- should not be locked twice
- should not be unlocked if it is not locked

drivers/usb/gadget/inode.c



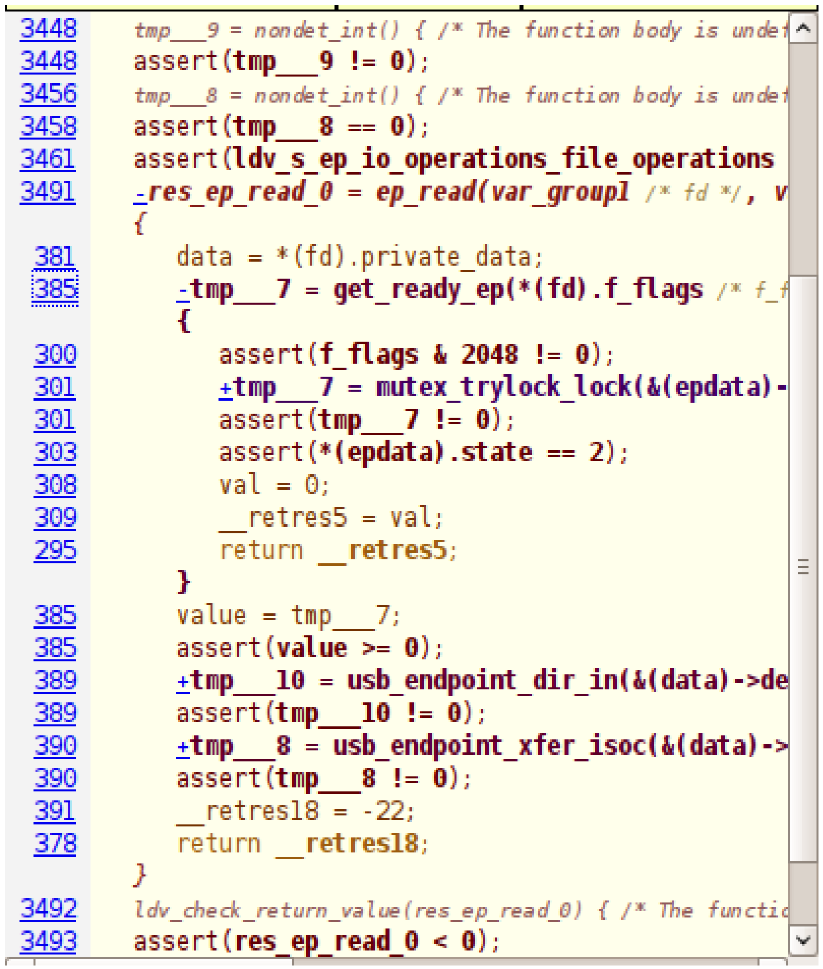
```
static ssize_t
ep_read (struct file *fd, char __user *buf, size_t len, loff_t *ptr)
{
	struct ep_data		*data = fd->private_data;
	void			*kbuf;
	ssize_t			value;

	if ((value = get_ready_ep (fd->f_flags, data)) < 0)
		return value;

	/* halt any endpoint by doing a "wrong direction" i/o call */
	if (usb_endpoint_dir_in(&data->desc)) {
		if (usb_endpoint_xfer_isoc(&data->desc))
			return -EINVAL;
		DBG (data->dev, "%s halt\n", data->name);
		spin_lock_irq (&data->dev->lock);
		if (likely (data->ep != NULL))
			usb_ep_set_halt (data->ep);
		spin_unlock_irq (&data->dev->lock);
		mutex_unlock(&data->lock);
		return -EBADMSG;
	}

	/* FIXME readahead for O_NONBLOCK and poll(); careful with ZLPs */

	value = -ENOMEM;
	kbuf = kmalloc (len, GFP_KERNEL);
	if (unlikely (!kbuf))
		goto free1;
```

drivers/usb/gadget/inode.c

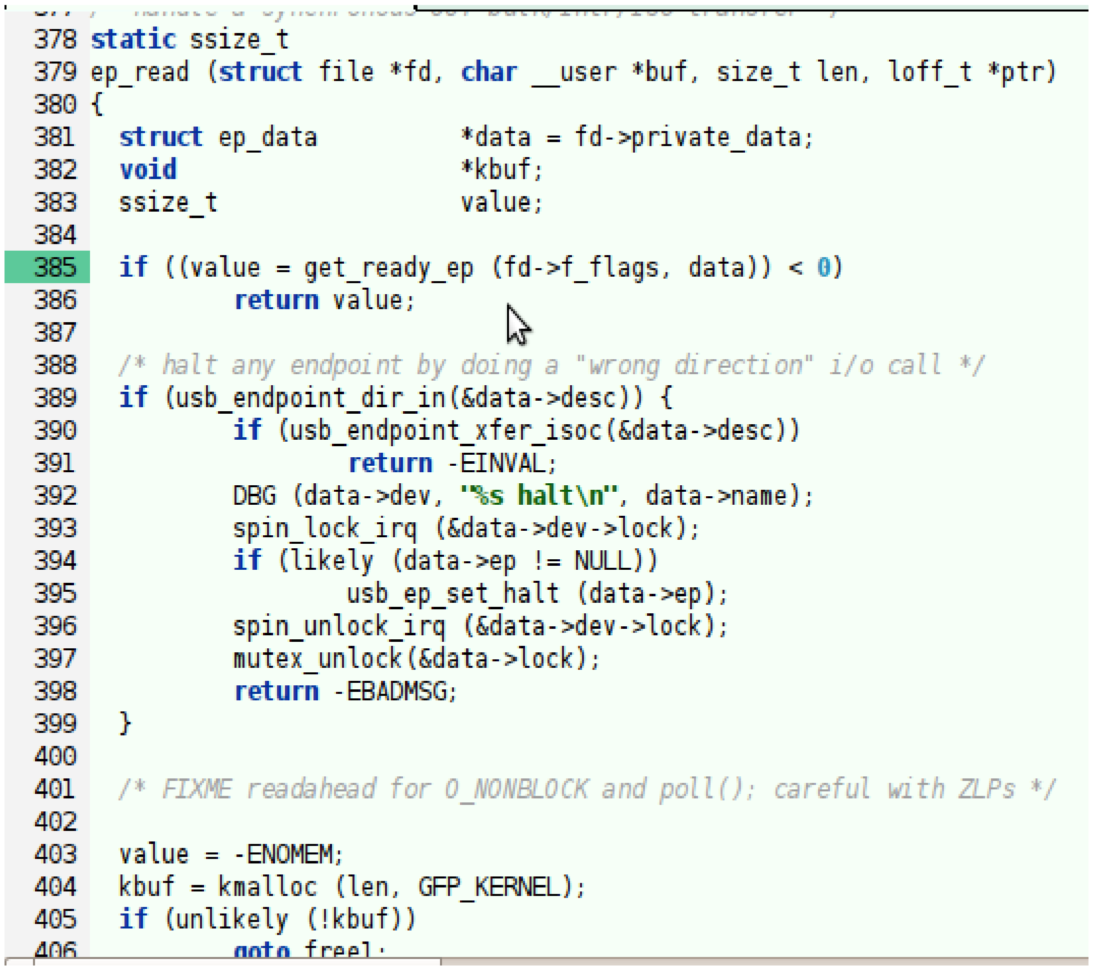


```c
static int
get_ready_ep (unsigned f_flags, struct ep_data *epdata)
{
        int     val;

        if (f_flags & O_NONBLOCK) {
                if (!mutex_trylock(&epdata->lock))
                        goto nonblock;
                if (epdata->state != STATE_EP_ENABLED) {
                        mutex_unlock(&epdata->lock);
nonblock:
                        val = -EAGAIN;
                } else
                        val = 0;
                return val;
        }

        val = mutex_lock_interruptible(&epdata->lock);
        if (val < 0)
                return val;

        switch (epdata->state) {
        case STATE_EP_ENABLED:
                break;
        case STATE_EP_UNBOUND:                  /* clean disconnect */
                val = -ENODEV;
                mutex_unlock(&epdata->lock);
        }
        return val;
}
```

drivers/usb/gadget/inode.c



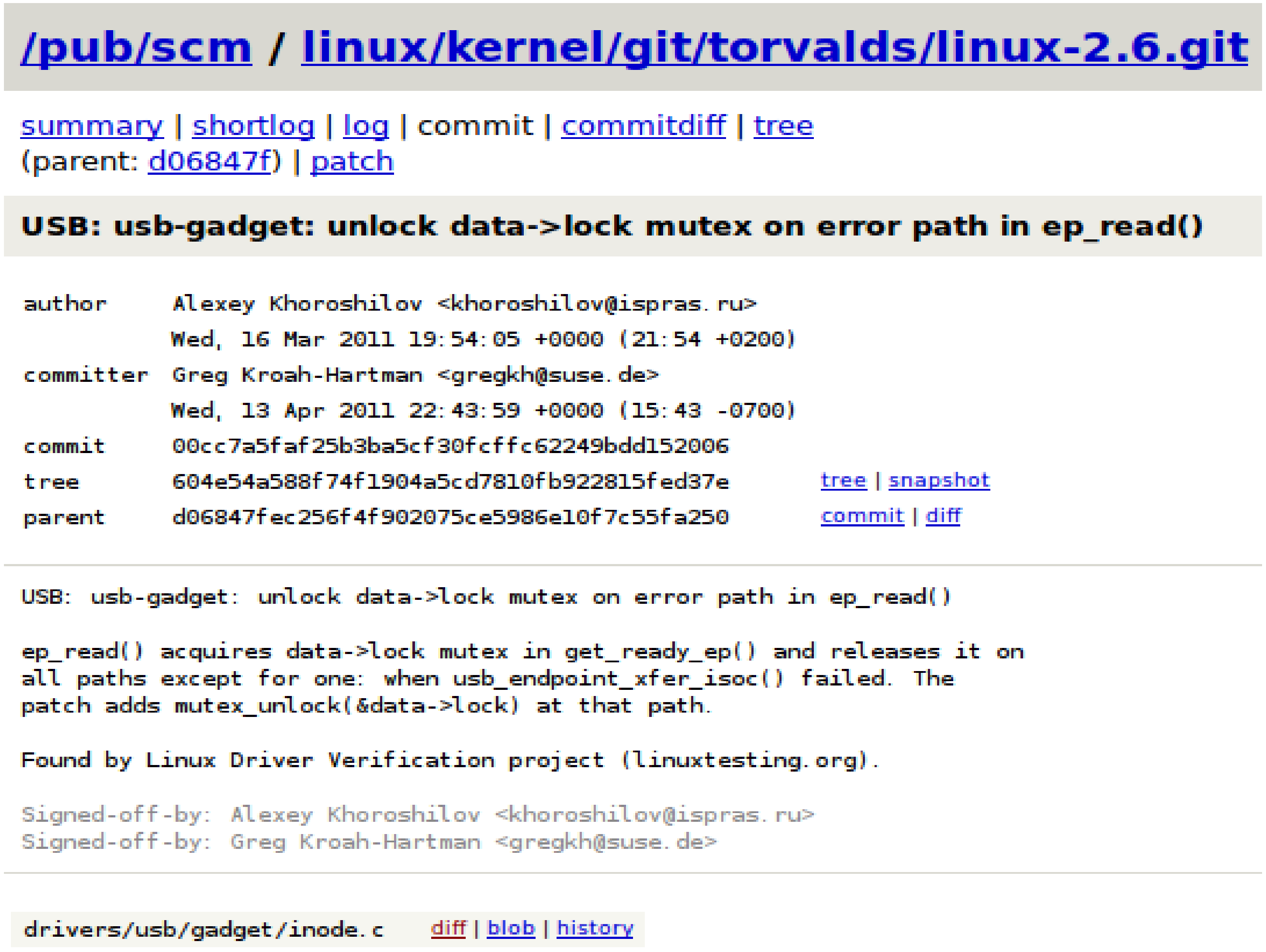
```
3448    tmp___9 = nondet_int() { /* The function body is undef
3448    assert(tmp___9 != 0);
3456    tmp___8 = nondet_int() { /* The function body is undef
3458    assert(tmp___8 == 0);
3461    assert(ldv_s_ep_io_operations_file_operations
3491    -res_ep_read_0 = ep_read(var_group1 /* fd */, v
        {
 381        data = *(fd).private_data;
 385        -tmp___7 = get_ready_ep(*(fd).f_flags /* f_f
            {
 300            assert(f_flags & 2048 != 0);
 301            +tmp___7 = mutex_trylock_lock(&(epdata)-
 301            assert(tmp___7 != 0);
 303            assert(*(epdata).state == 2);
 308            val = 0;
 309            __retres5 = val;
 295            return __retres5;
            }
 385        value = tmp___7;
 385        assert(value >= 0);
 389        +tmp___10 = usb_endpoint_dir_in(&(data)->de
 389        assert(tmp___10 != 0);
 390        +tmp___8 = usb_endpoint_xfer_isoc(&(data)->
 390        assert(tmp___8 != 0);
 391        __retres18 = -22;
 378        return __retres18;
        }
3492    ldv_check_return_value(res_ep_read_0) { /* The functio
3493    assert(res_ep_read_0 < 0);
```



```
378 static ssize_t
379 ep_read (struct file *fd, char __user *buf, size_t len, loff_t *ptr)
380 {
381     struct ep_data          *data = fd->private_data;
382     void                    *kbuf;
383     ssize_t                 value;
384 
385     if ((value = get_ready_ep (fd->f_flags, data)) < 0)
386             return value;
387 
388     /* halt any endpoint by doing a "wrong direction" i/o call */
389     if (usb_endpoint_dir_in(&data->desc)) {
390             if (usb_endpoint_xfer_isoc(&data->desc))
391                     return -EINVAL;
392             DBG (data->dev, "%s halt\n", data->name);
393             spin_lock_irq (&data->dev->lock);
394             if (likely (data->ep != NULL))
395                     usb_ep_set_halt (data->ep);
396             spin_unlock_irq (&data->dev->lock);
397             mutex_unlock(&data->lock);
398             return -EBADMSG;
399     }
400 
401     /* FIXME readahead for O_NONBLOCK and poll(); careful with ZLPs */
402 
403     value = -ENOMEM;
404     kbuf = kmalloc (len, GFP_KERNEL);
405     if (unlikely (!kbuf))
406             goto free1;
```

LinuxTesting.org

drivers/usb/gadget/inode.c



/pub/scm / linux/kernel/git/torvalds/linux-2.6.git

summary | shortlog | log | commit | commitdiff | tree
(parent: d06847f) | patch

USB: usb-gadget: unlock data->lock mutex on error path in ep_read()

```
author     Alexey Khoroshilov <khoroshilov@ispras.ru>
           Wed, 16 Mar 2011 19:54:05 +0000 (21:54 +0200)
committer  Greg Kroah-Hartman <gregkh@suse.de>
           Wed, 13 Apr 2011 22:43:59 +0000 (15:43 -0700)
commit     00cc7a5faf25b3ba5cf30fcffc62249bdd152006
tree       604e54a588f74f1904a5cd7810fb922815fed37e      tree | snapshot
parent     d06847fec256f4f902075ce5986e10f7c55fa250      commit | diff
```

```
USB: usb-gadget: unlock data->lock mutex on error path in ep_read()

ep_read() acquires data->lock mutex in get_ready_ep() and releases it on
all paths except for one: when usb_endpoint_xfer_isoc() failed. The
patch adds mutex_unlock(&data->lock) at that path.

Found by Linux Driver Verification project (linuxtesting.org).

Signed-off-by: Alexey Khoroshilov <khoroshilov@ispras.ru>
Signed-off-by: Greg Kroah-Hartman <gregkh@suse.de>
```

drivers/usb/gadget/inode.c   diff | blob | history

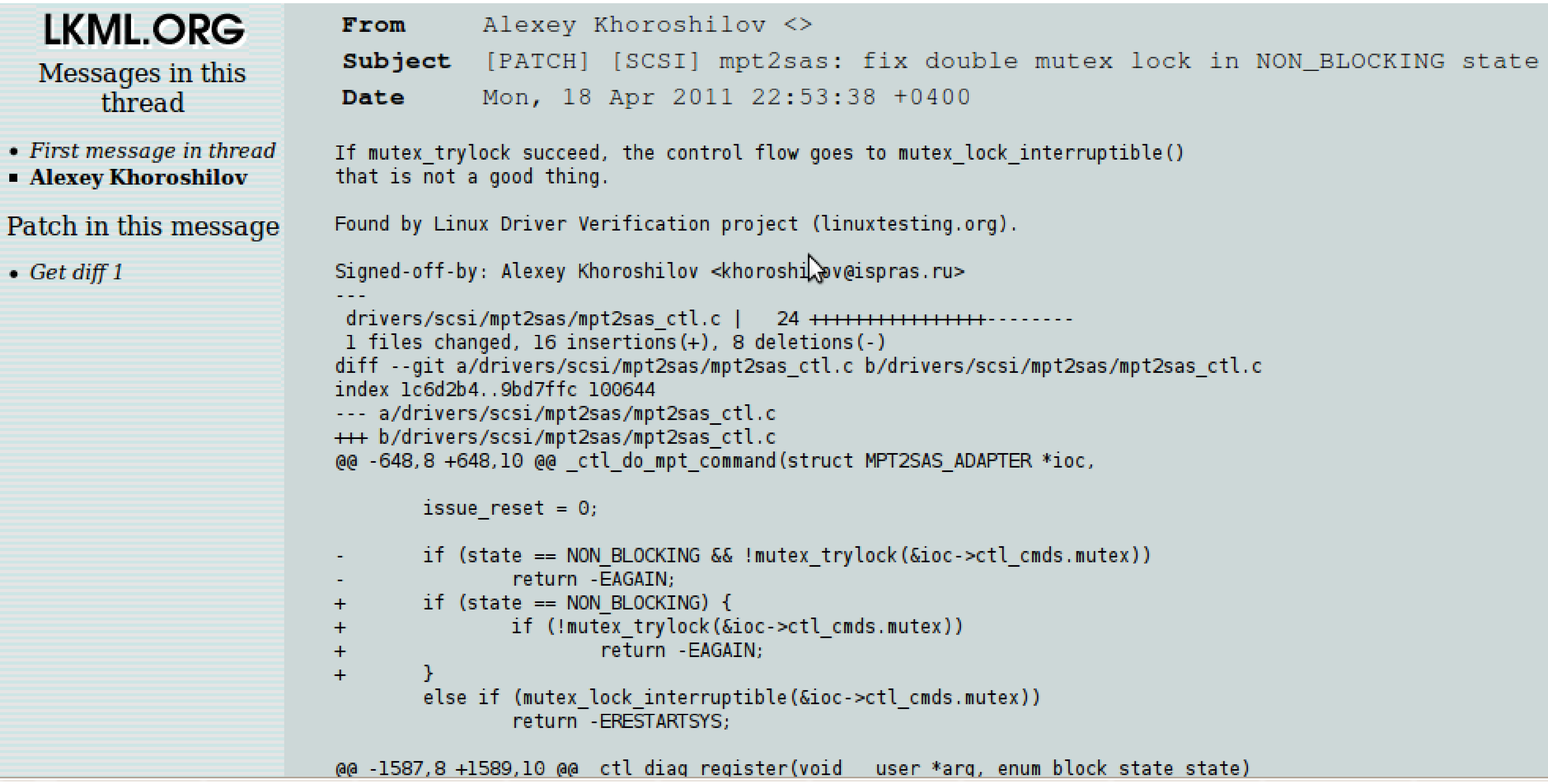
Coccinelle

- Intra-procedural analysis
- Limited data-flow analysis

drivers/scsi/mpt2sas/mpt2sas_ctl.c

```
/**
 * _ctl_do_mpt_command - main handler for MPT2COMMAND  opcode
 * @state - NON_BLOCKING or BLOCKING
 */
static long
_ctl_do_mpt_command(...) {
...
        if (state == NON_BLOCKING && !mutex_trylock(&ioc->ctl_cmds.mutex))
                return -EAGAIN;
        else if (mutex_lock_interruptible(&ioc->ctl_cmds.mutex))
                return -ERESTARTSYS;

```

drivers/scsi/mpt2sas/mpt2sas_ctl.c



LKML.ORG

Messages in this thread

- *First message in thread*
  - **Alexey Khoroshilov**

Patch in this message

- *Get diff 1*

```
From      Alexey Khoroshilov <>
Subject   [PATCH] [SCSI] mpt2sas: fix double mutex lock in NON_BLOCKING state
Date      Mon, 18 Apr 2011 22:53:38 +0400

If mutex_trylock succeed, the control flow goes to mutex_lock_interruptible()
that is not a good thing.

Found by Linux Driver Verification project (linuxtesting.org).

Signed-off-by: Alexey Khoroshilov <khoroshilov@ispras.ru>
---
 drivers/scsi/mpt2sas/mpt2sas_ctl.c |   24 ++++++++++++++++--------
 1 files changed, 16 insertions(+), 8 deletions(-)
diff --git a/drivers/scsi/mpt2sas/mpt2sas_ctl.c b/drivers/scsi/mpt2sas/mpt2sas_ctl.c
index 1c6d2b4..9bd7ffc 100644
--- a/drivers/scsi/mpt2sas/mpt2sas_ctl.c
+++ b/drivers/scsi/mpt2sas/mpt2sas_ctl.c
@@ -648,8 +648,10 @@ _ctl_do_mpt_command(struct MPT2SAS_ADAPTER *ioc,

        issue_reset = 0;

-       if (state == NON_BLOCKING && !mutex_trylock(&ioc->ctl_cmds.mutex))
-               return -EAGAIN;
+       if (state == NON_BLOCKING) {
+               if (!mutex_trylock(&ioc->ctl_cmds.mutex))
+                       return -EAGAIN;
+       }
        else if (mutex_lock_interruptible(&ioc->ctl_cmds.mutex))
                return -ERESTARTSYS;

@@ -1587,8 +1589,10 @@ _ctl_diag_register(void __user *arg, enum block_state state)
```

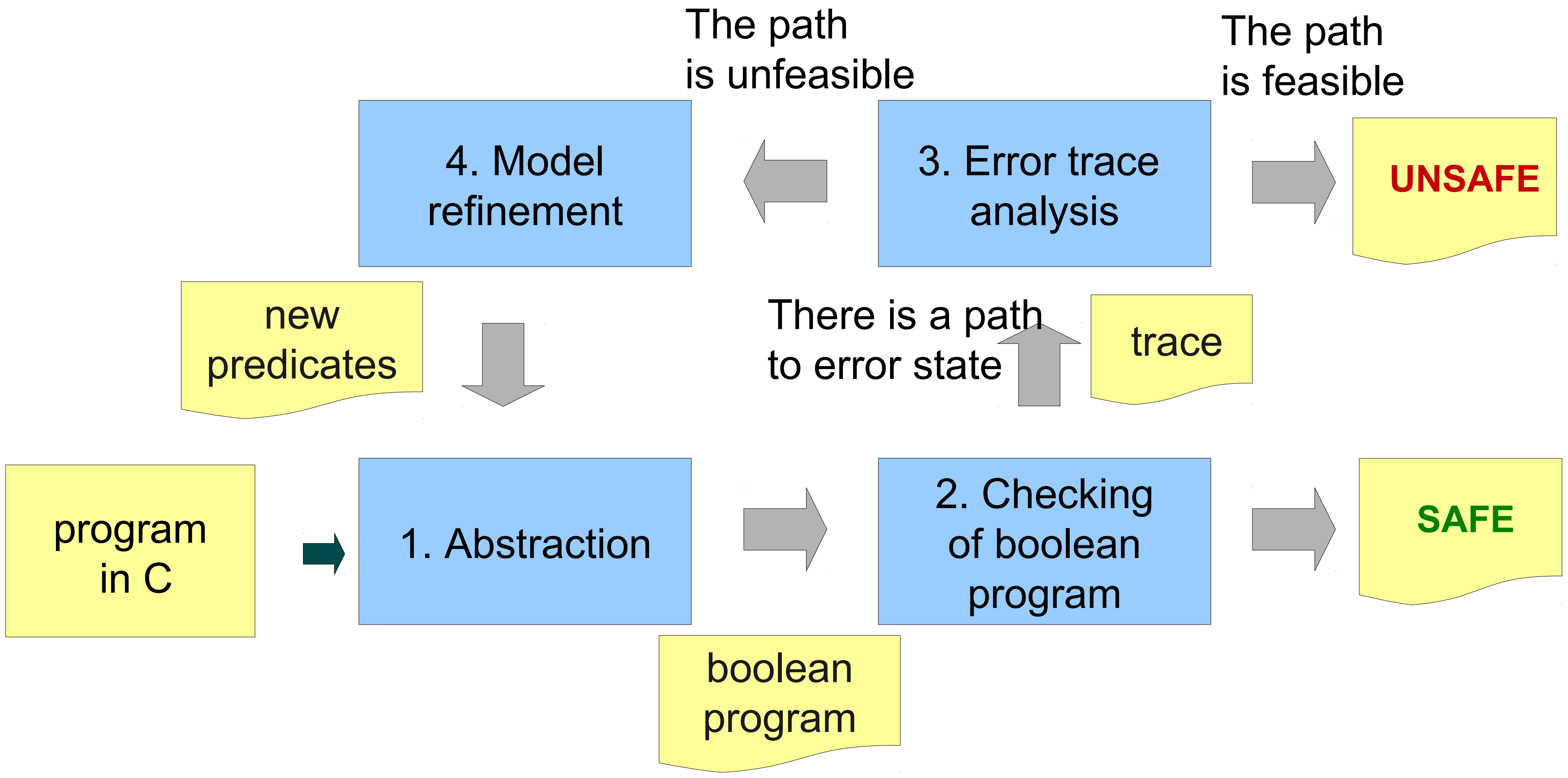
Heavy-Weight Analysis

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

How it works?

- CEGAR - **C**ounter-**E**xample **G**uided **A**bstraction **R**efinement

CEGAR


The path is unfeasible
The path is feasible
4. Model refinement
3. Error trace analysis
UNSAFE
new predicates
There is a path to error state
trace
program in C
1. Abstraction
2. Checking of boolean program
SAFE
boolean program

CEGAR-based Heavy-Weight Tools

Commercial:

- **Microsoft SDV**

Academic:

- BLAST
- CPAChecker (U. Passau)
- SATABS (U. Oxford)
- ARMC (U. Munich)

Microsoft Static Driver Verifier

We've created a number of things to do rich static analysis. We actually went out and **bought for a little over $30 million a company** that was in the business of building those kinds of tools, and we said now we want you to focus on applying these tools to large-scale software systems, **the kind of system we have in the source code of Windows or Office**, and see how far we can get on this.

……

We call the system that does this kind of proof, it's a model-checking system. You describe the constraints, including things as simple as nobody should acquire the lock if they've already acquired it, nobody should release it if they haven't acquired it, certain things about the multi-threading aspect of the code that you want to make sure work very well. And you describe those things literally, in this case in the C code itself, and then the analyzer goes through and reduces the program, takes away anything that doesn't affect the path analysis that it's trying to go through to determine is there some path through the program that violates the constraints.

The initial domain we applied this in was in device drivers.

Bill Gates at
17th Annual ACM Conference on Object-Oriented Programming, Systems, Languages and Application, 2002

Microsoft Static Driver Verifier

- Included into Microsoft Windows Driver Developer Kit (DDK) in 2006
- Continuous improvements:
  - Kinds of interfaces:
    WDM (2006) → WDM, NDIS, KMDF (2010)
  - Number of rules:
    43 (2006) → 200 (2010)
  - Time required to analyze one driver:
    ???→ 2-3 hours (2010)

Microsoft Static Driver Verifier

Results

- 33 critical bugs in the WDK sample drivers
- 53 critical bugs in kernel-mode drivers

CEGAR-based Heavy-Weight Tools

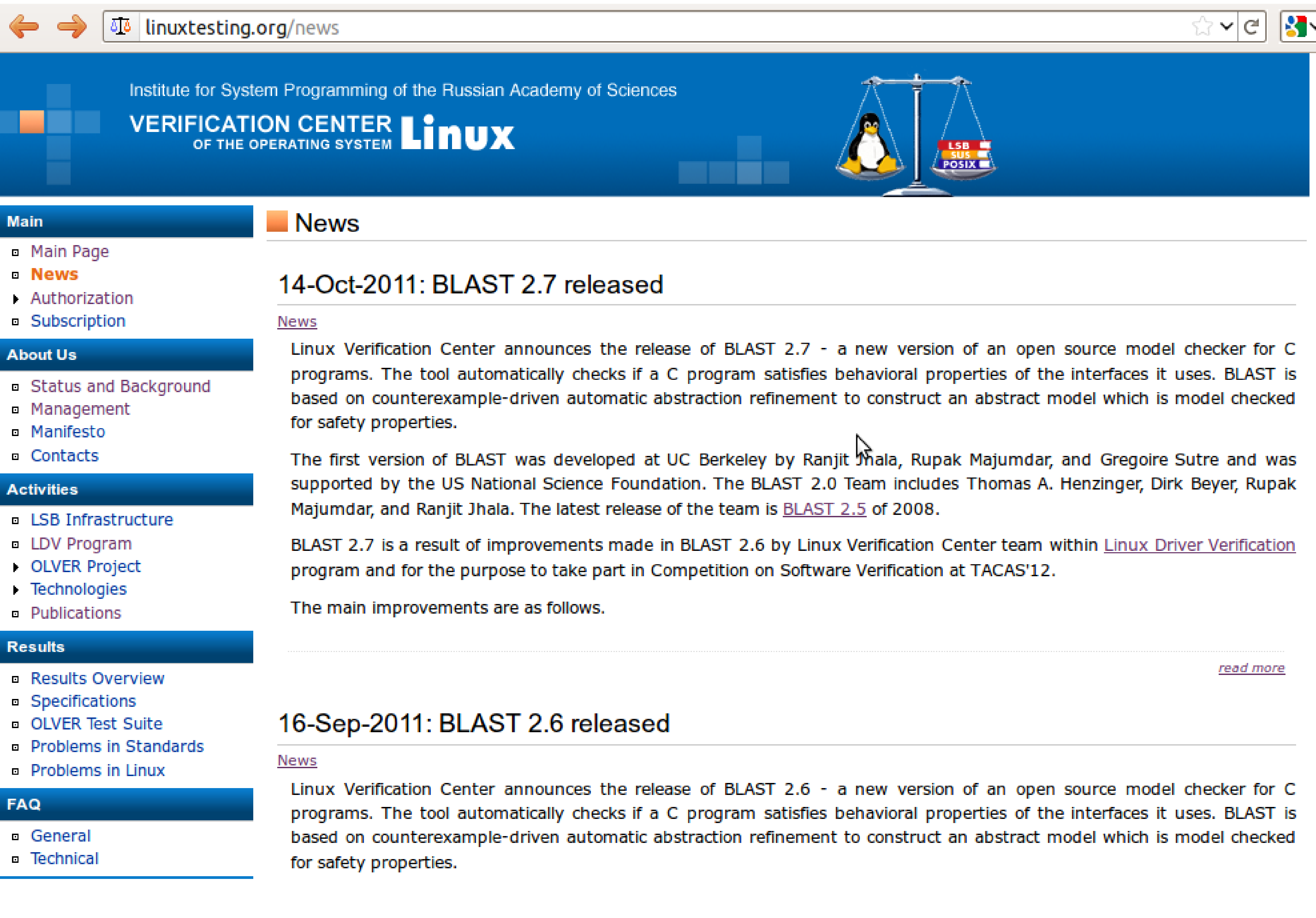
Commercial:

- Microsoft SDV

Academic:

- **BLAST**
- **CPAChecker** (U. Passau)
- SATABS (U. Oxford)
- ARMC (U. Munich)

LinuxTesting.org



News

14-Oct-2011: BLAST 2.7 released

News

Linux Verification Center announces the release of BLAST 2.7 - a new version of an open source model checker for C programs. The tool automatically checks if a C program satisfies behavioral properties of the interfaces it uses. BLAST is based on counterexample-driven automatic abstraction refinement to construct an abstract model which is model checked for safety properties.

The first version of BLAST was developed at UC Berkeley by Ranjit Jhala, Rupak Majumdar, and Gregoire Sutre and was supported by the US National Science Foundation. The BLAST 2.0 Team includes Thomas A. Henzinger, Dirk Beyer, Rupak Majumdar, and Ranjit Jhala. The latest release of the team is BLAST 2.5 of 2008.

BLAST 2.7 is a result of improvements made in BLAST 2.6 by Linux Verification Center team within Linux Driver Verification program and for the purpose to take part in Competition on Software Verification at TACAS'12.

The main improvements are as follows.

read more

16-Sep-2011: BLAST 2.6 released

News

Linux Verification Center announces the release of BLAST 2.6 - a new version of an open source model checker for C programs. The tool automatically checks if a C program satisfies behavioral properties of the interfaces it uses. BLAST is based on counterexample-driven automatic abstraction refinement to construct an abstract model which is model checked for safety properties.

Yet another static analysis tool?

Linux Driver Verification Program

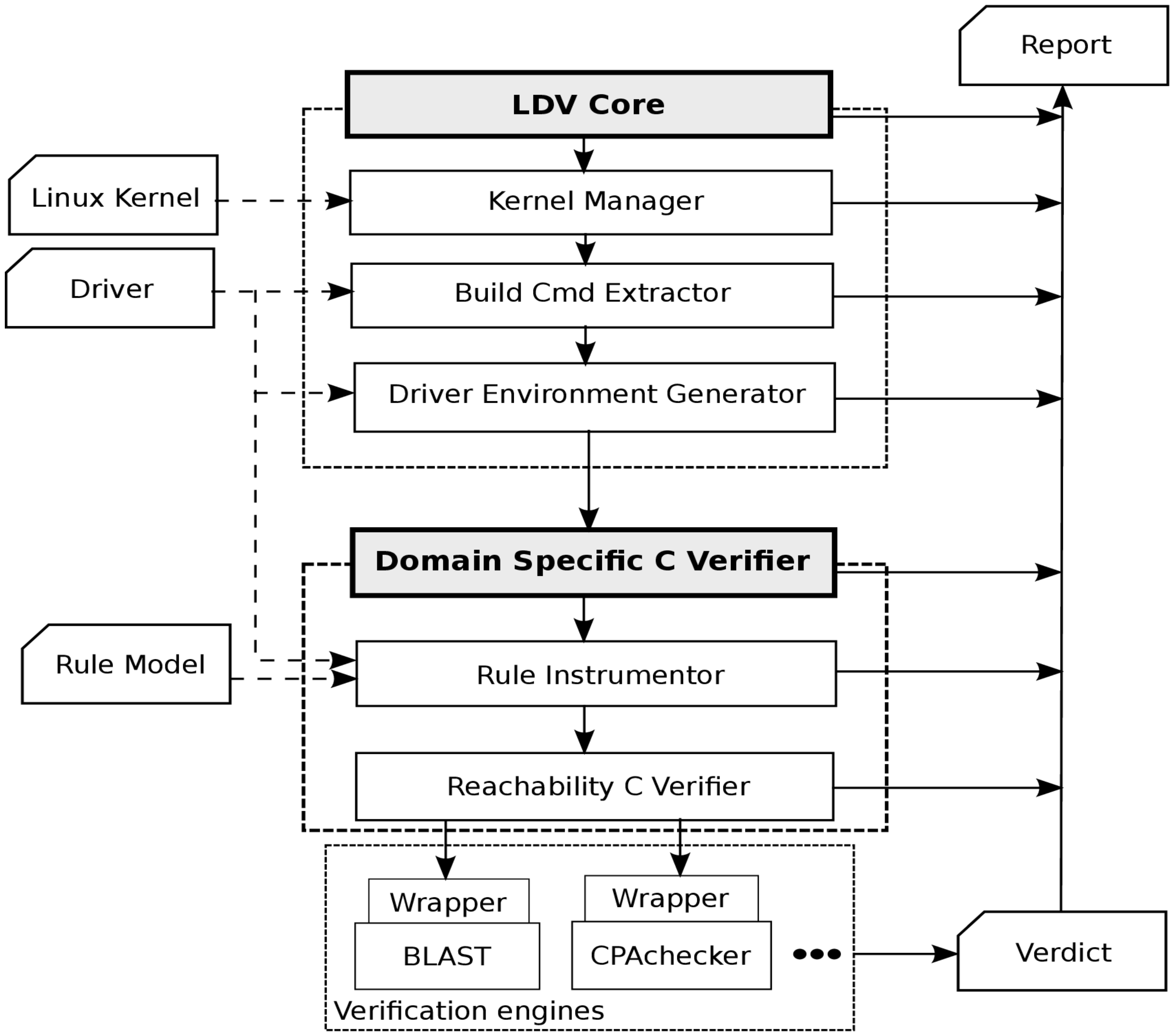
- Yes, our idea is to promote heavy-weight verification tools
- But our idea is **NOT** to push a particular verification technique

LDV Goals

- Provide infrastructure for application of verification tools to Linux device drivers
- Research new verification approaches in the industrial settings
- Improve quality of the Linux device drivers
- Provide a basis for education of young researches

Where we are

- Static analysis infrastructure



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

Verification Tools World

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

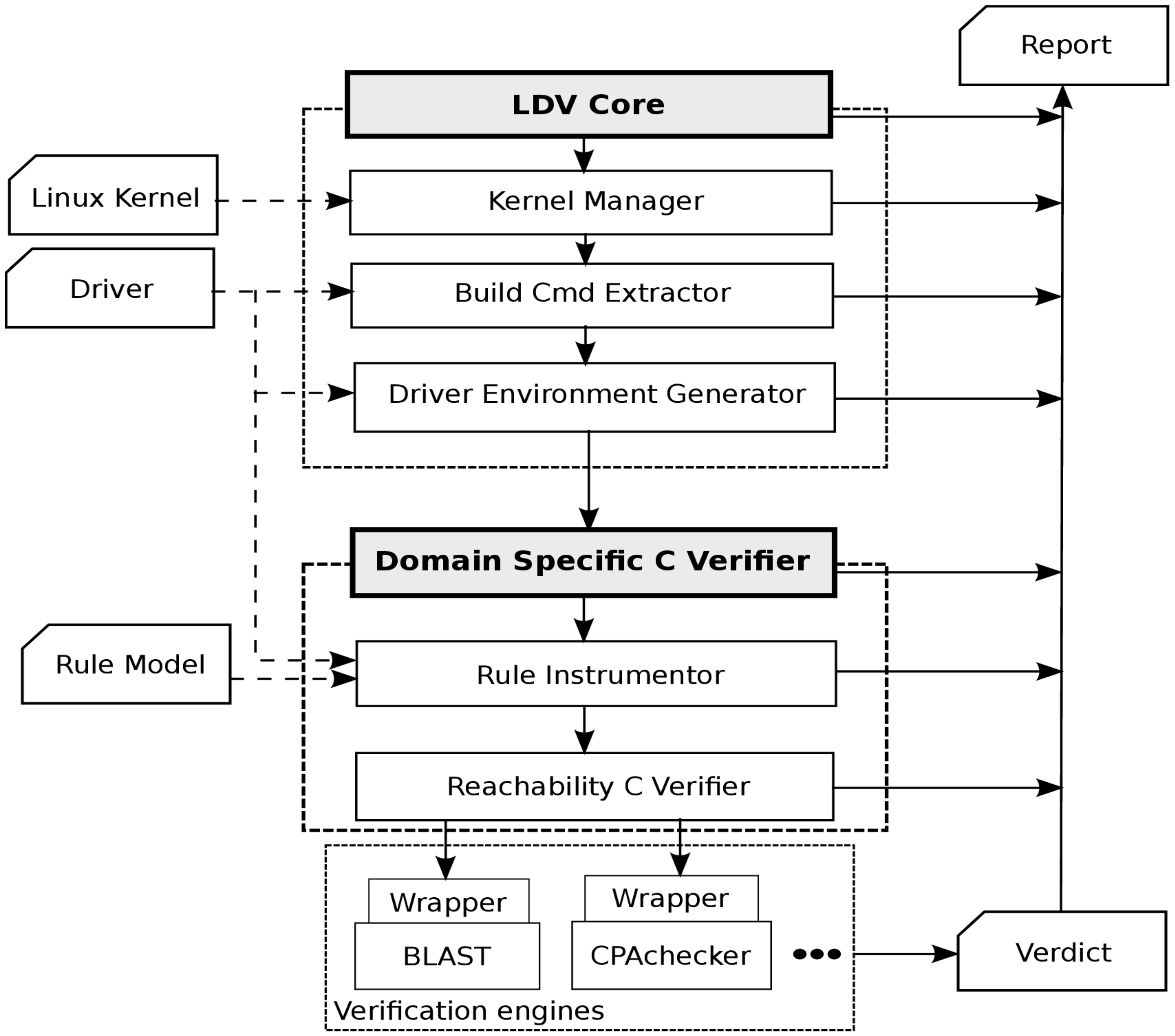
```
void other_func(int v)
{
    ...
    assert( x != NULL);
}
```

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


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
Wrapper
BLAST
CPAchecker
•••
Verdict
Verification engines

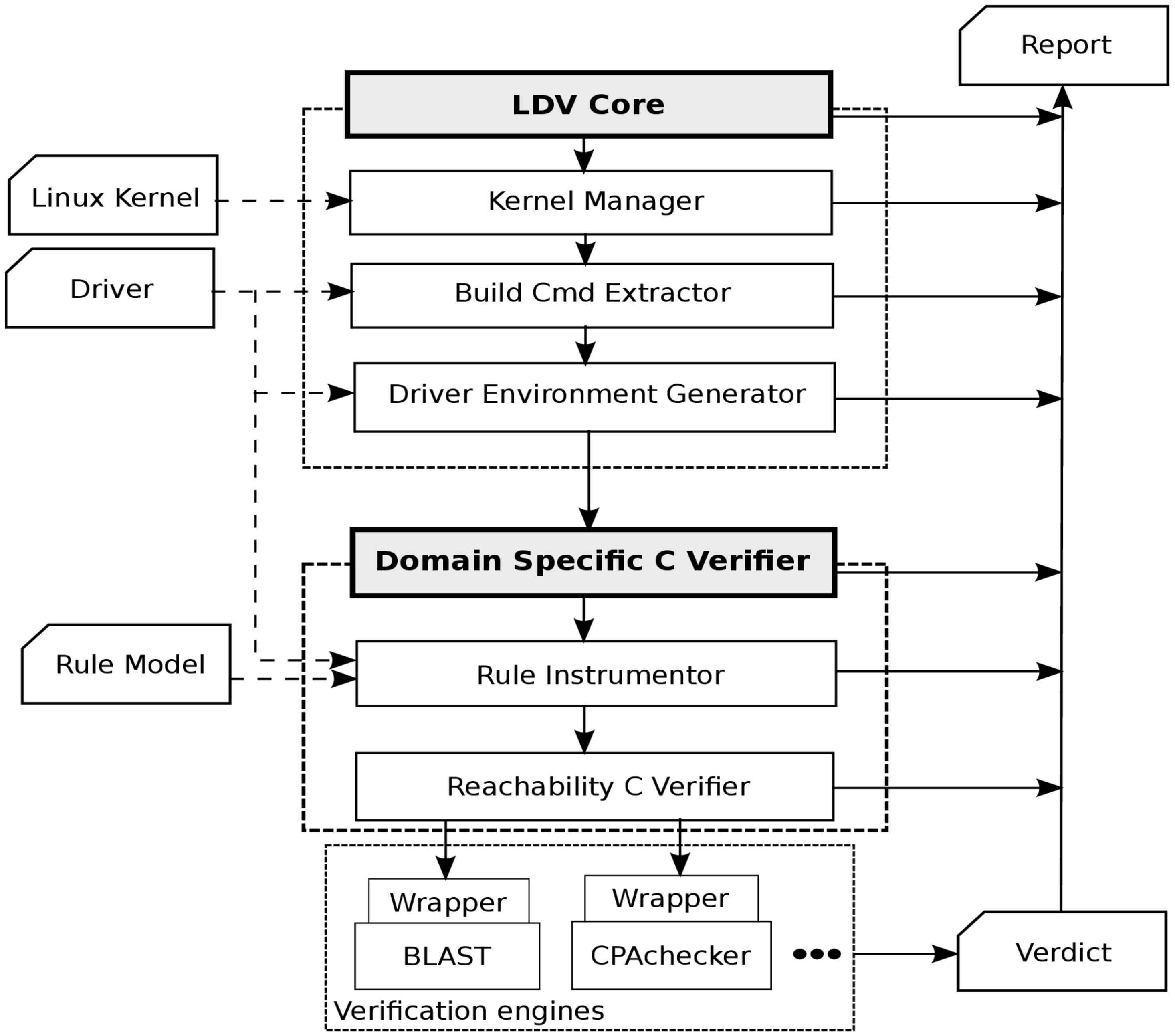
LinuxTesting.org

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;
}
```

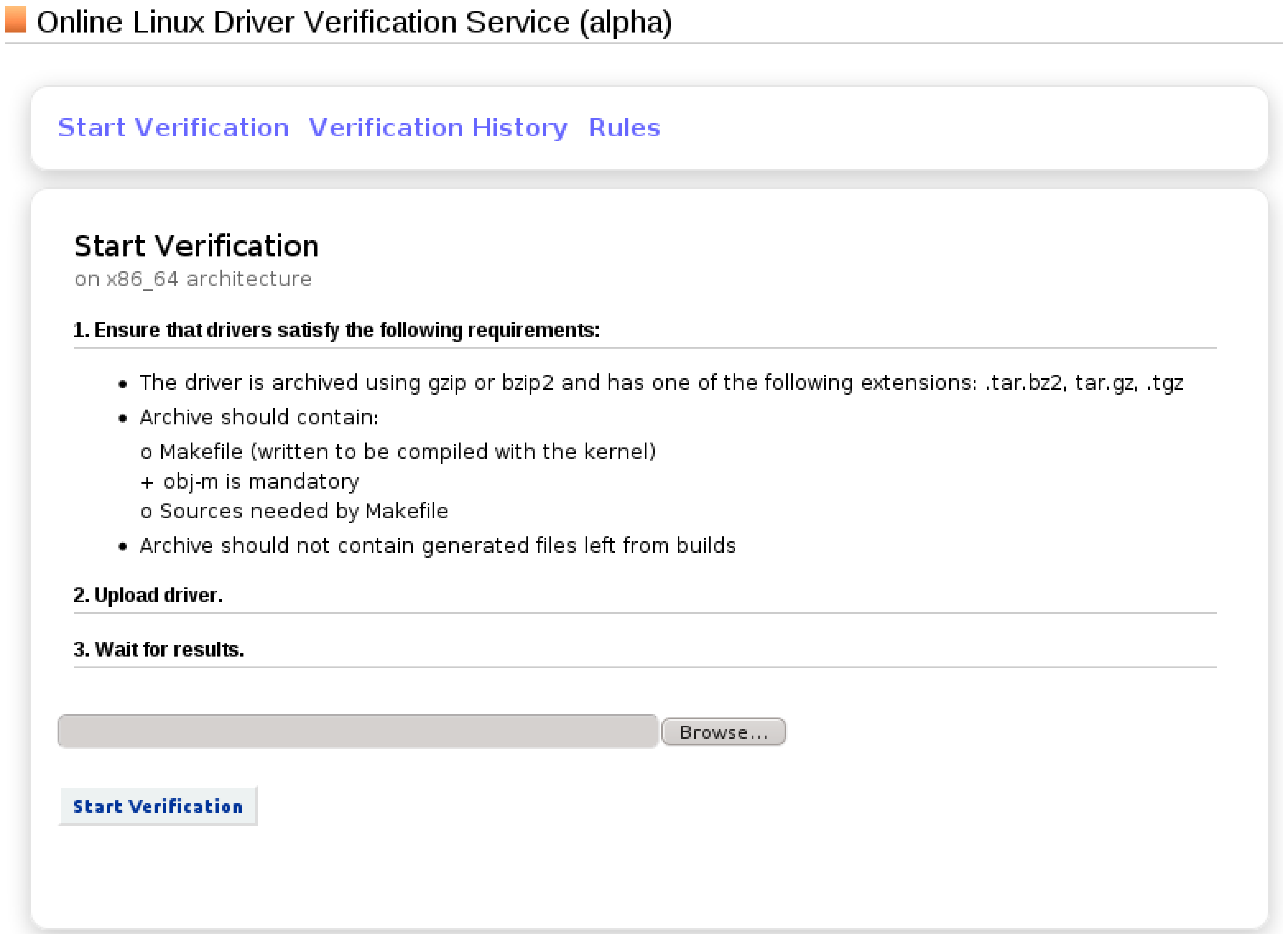

Report
LDV Core
Linux Kernel
Kernel Manager
Driver
Build Cmd Extractor
Driver Environment Generator
Domain Specific C Verifier
Rule Model
Rule Instrumentor
Reachability C Verifier
Wrapper
Wrapper
BLAST
CPAchecker
Verdict
Verification engines

Where we are

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

LinuxTesting.org

ldv-online



Online Linux Driver Verification Service (alpha)

Start Verification

on x86_64 architecture

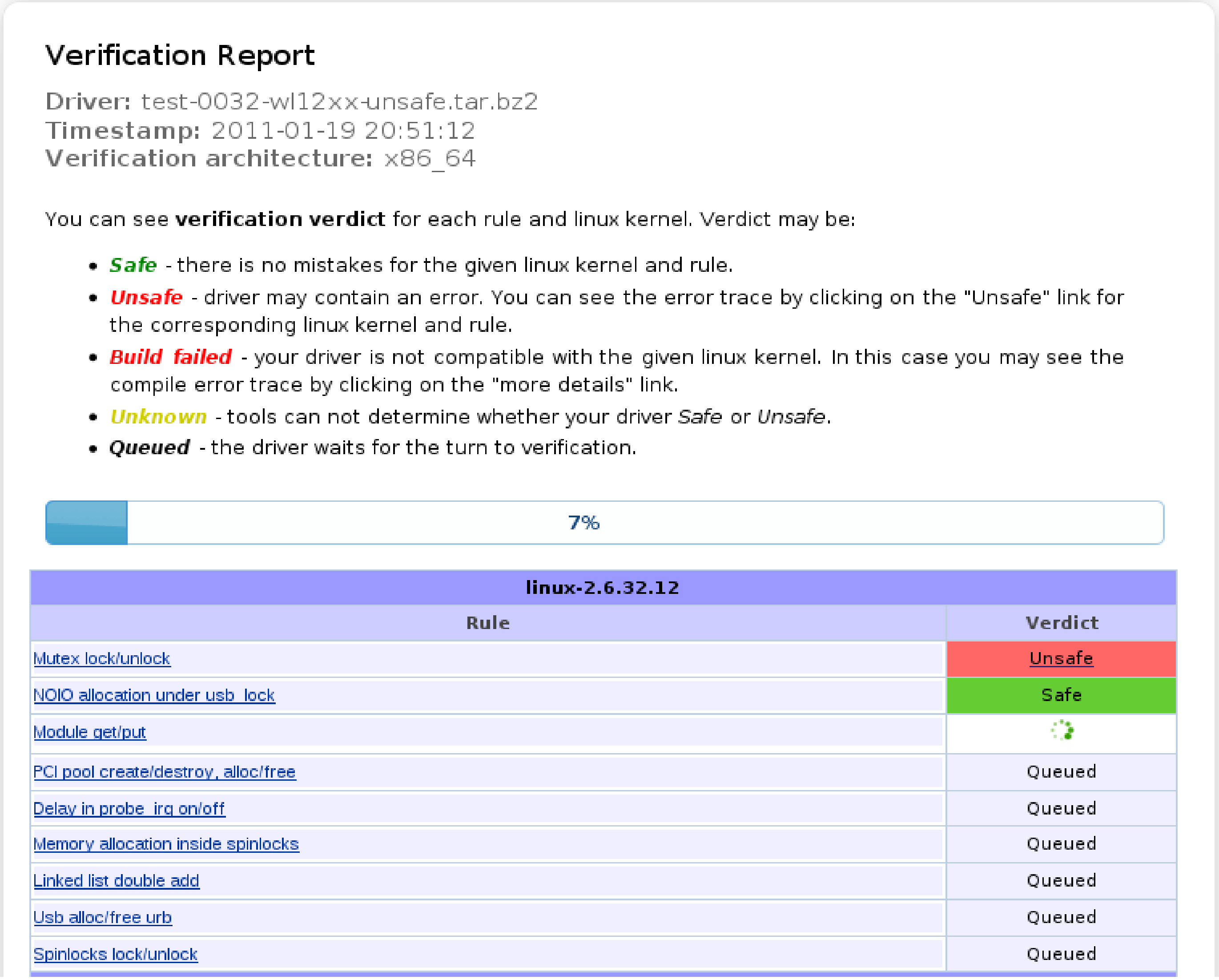
1. Ensure that drivers satisfy the following requirements:

- 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.

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%

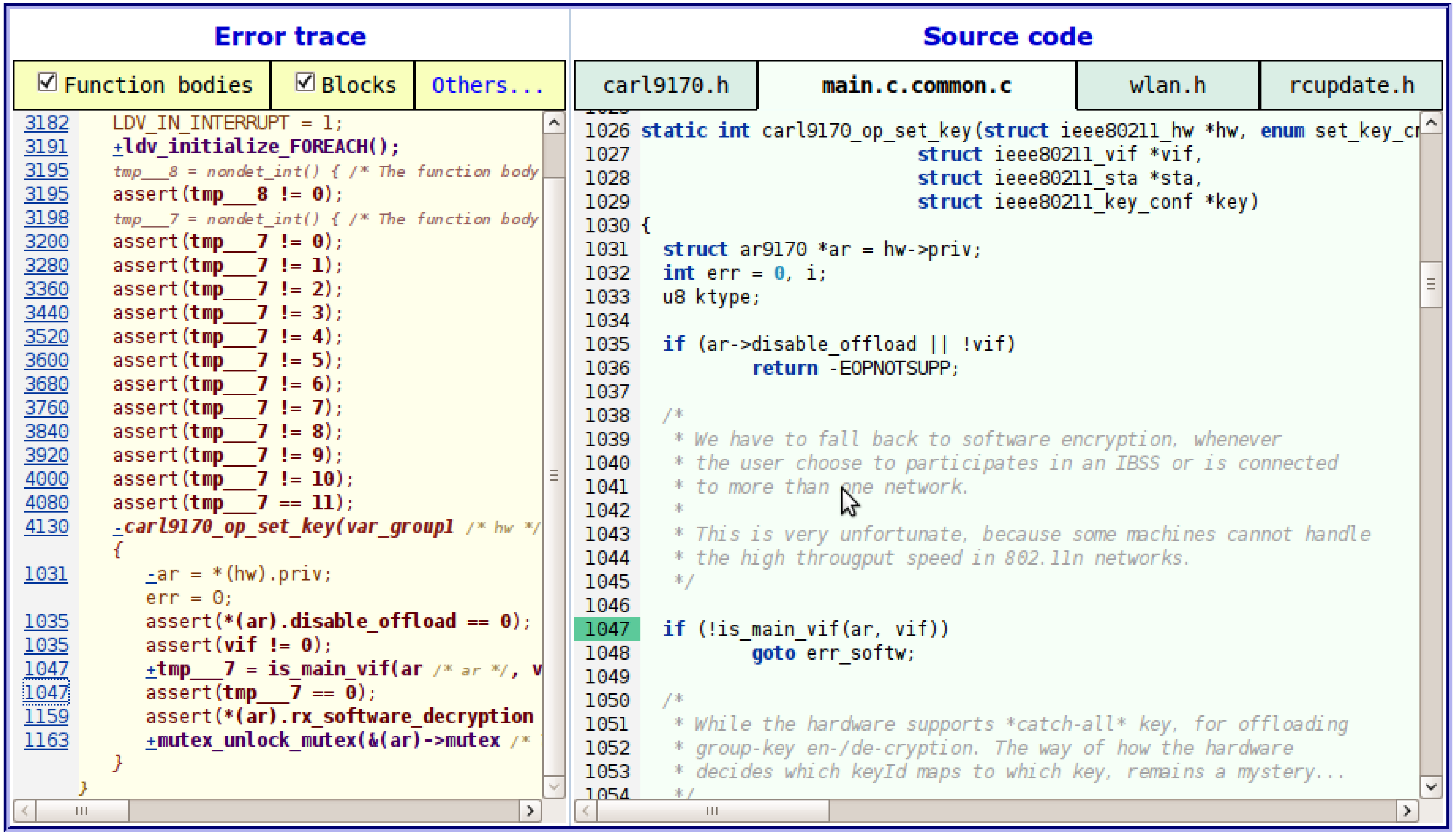
linux-2.6.32.12	
Rule	**Verdict**
Mutex lock/unlock	Unsafe
NOIO allocation under usb_lock	Safe
Module get/put	
PCI pool create/destroy, alloc/free	Queued
Delay in probe_irq on/off	Queued
Memory allocation inside spinlocks	Queued
Linked list double add	Queued
Usb alloc/free urb	Queued
Spinlocks lock/unlock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

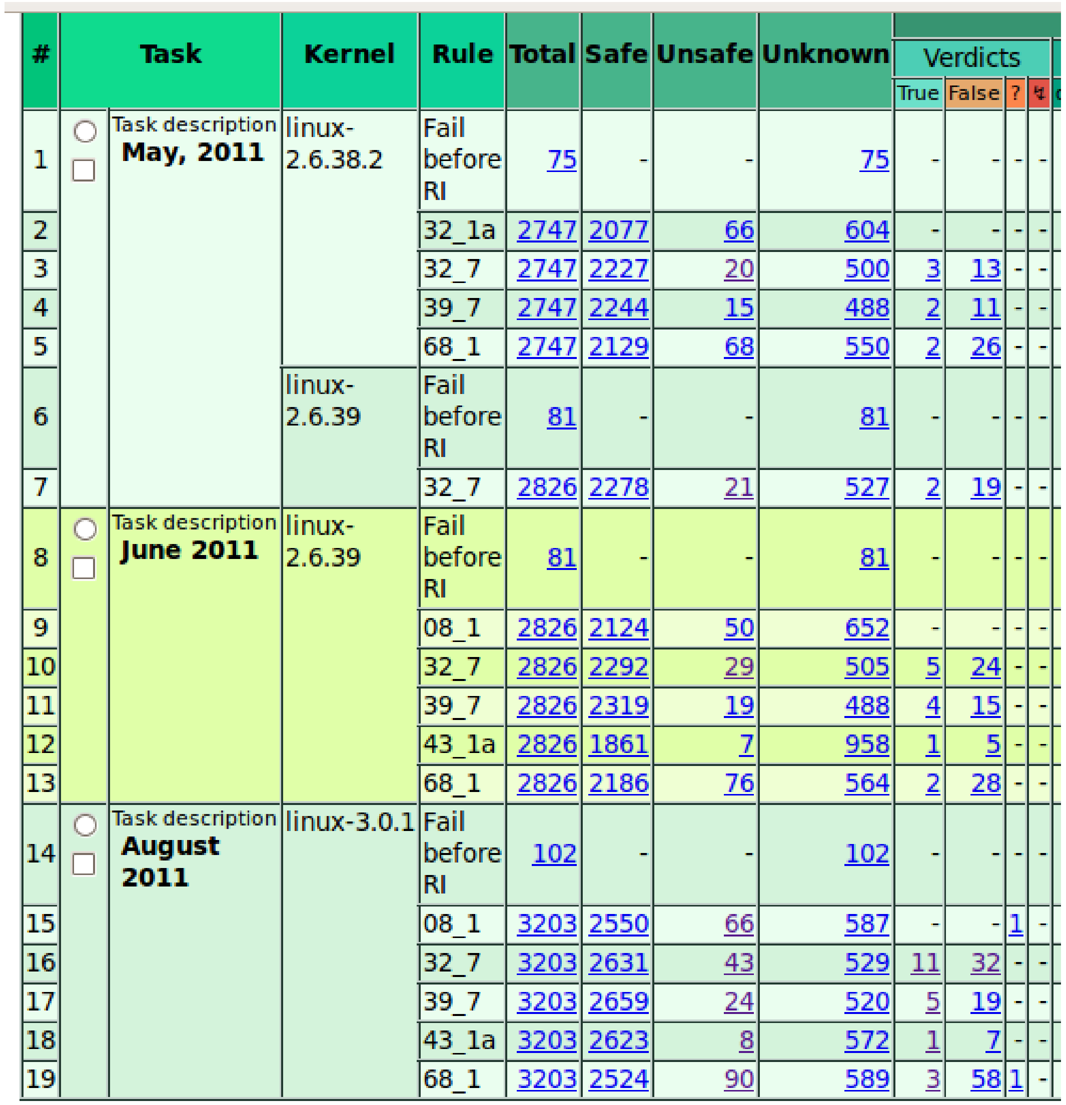
Rule: Mutex lock/unlock



```
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 /*
         }
     }
```

```
1026 static int carl9170_op_set_key(struct ieee80211_hw *hw, enum set_key_cı
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



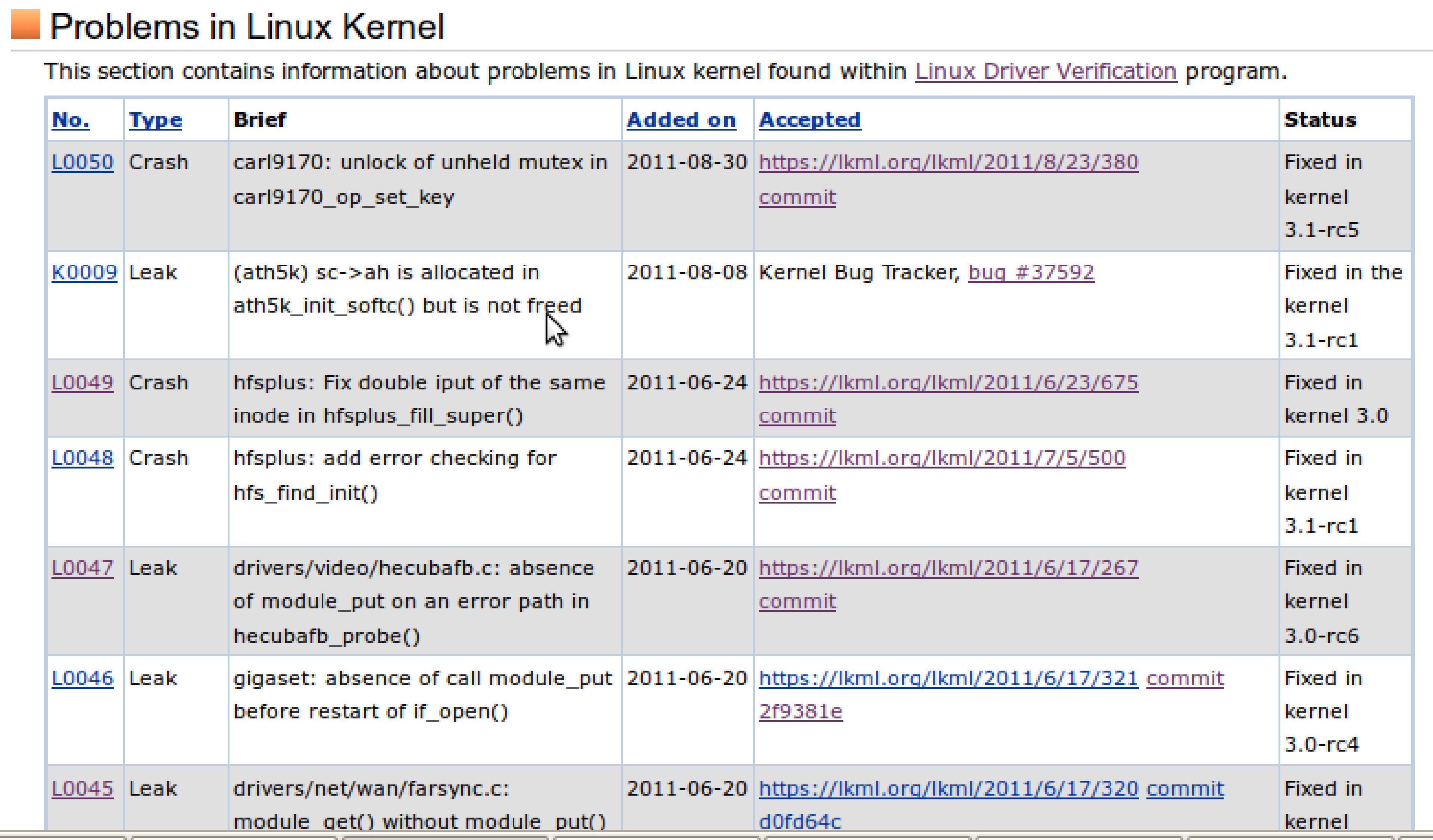
#	Task		Kernel	Rule	Total	Safe	Unsafe	Unknown	Verdicts			
									True	False	?	↯
1	○ ☐	Task description **May, 2011**	linux-2.6.38.2	Fail before RI	75	-	-	75	-	-	-	-
2				32_1a	2747	2077	66	604	-	-	-	-
3				32_7	2747	2227	20	500	3	13	-	-
4				39_7	2747	2244	15	488	2	11	-	-
5				68_1	2747	2129	68	550	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	-

LinuxTesting.org

Bugs Found

http://linuxtesting.org/results/ldv

- 42 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

Where we are

but there is no magic

- Verification tools
    - issues with pointer analysis, container_of, functional_pointers, complex data structures
- Environment generator
    - issues with inaccurate environment model in some cases
- RuleDB
    - only 5 rules formalized and debugged

Where we are going

- **Improve verification tools**
- **Formalize new rules**
- Continuous application of the tools to Linux device drivers
- Integrate new verification tools

What we are looking for

- Prioritization of rules
- Identification of new rules
- Industrial partners
- Computational power

Conlusions

- Heavy-weight verification is useful in practice
- LDV infrastructure is ready for research and industrial usage
- Number of supported rules must be increased
- Help on rules prioritization and identification are welcome

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