Linux Qualification -
Coding Style / Type issues in IEC 61508

Nicholas Mc Guire <safety@osadl.org>

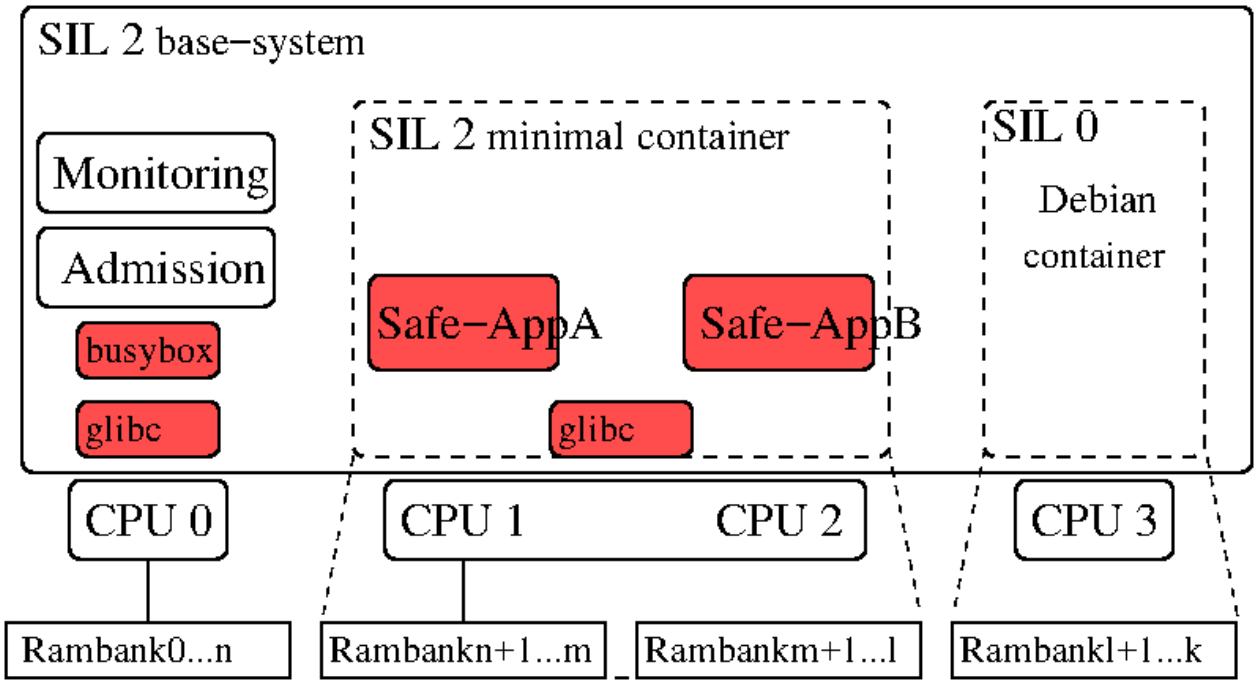
December 1, 2016

Outline

- SIL2LinuxMP Context
- Coding style/coding standard ?
- Short glimpse in the horror cabinet of Linux kernel code
- Type inconsistencies - the first real challenge
- Conclusion

Nicholas Mc Guire
<safety@osadl.

Context: system components



SIL2: kernel+glibc+busybox+safety_application

Coding Style

- Mandated but not defined
- Coding style and coding standards
- What is resonable ?

Nicholas Mc Guire
<safety@osadl.

How does Linux kernel do it ?

- CodingStyle - simple and relatively short (40+ rules)
- checkpatch.pl - exhaustive and fussy (400+ rules)
- amendment by tooling (sparse/coccinelle/checkpatch –strict) to cover some aspects that are not sufficiently addressable by coding style
- amendment by procedures (SubmittingPatches,SubmitChecklist)
- patch review procedure
- multi-layer integration process

So how good do we do in the kernel ?

reasonable conditions

drivers/media/dvb-frontends/dib7000m.c:926 bad conditional

```
/* P_dintl_native, P_dintlv_inv, P_hrch, P_code_rate, P_sel
value = 0;
if (1 != 0)
        value |= (1 << 6);
if (ch->hierarchy == 1)
        value |= (1 << 4);
if (1 == 1)
        value |= 1;
switch ((ch->hierarchy == 0 || 1 == 1) ?
        ch->code_rate_HP : ch->code_rate_LP) {
```


Nicholas Mc Guire <safety@osadl.

...and reasonable control flow

drivers/staging/rtl8723au/hal/rtl8723a_bt-coexist.c:7264 else duplicates if

```
...
} else if (maxInterval == 2) {
        btdm_2AntPsTdma(padapter, true, 15);
        pBtdm8723->psTdmaDuAdjType = 15;
} else if (maxInterval == 3) {
        btdm_2AntPsTdma(padapter, true, 15);
        pBtdm8723->psTdmaDuAdjType = 15;
} else {
        btdm_2AntPsTdma(padapter, true, 15);
        pBtdm8723->psTdmaDuAdjType = 15;
}
```

...no conditions with side-effects

drivers/ide/cmd640.c:680 redundant logic expression with side-effect

```
if (inb(0xCF8) == 0x00 && inb(0xCF8) == 0x00) {
        spin_unlock_irqrestore(&cmd640_lock, flags);
        return 1;
}
```

This has been in here since kernel 2.3.X (predates git) The earlier 2.2.X kernels do not have this construct
How did this get into the kernel ?

Nicholas Mc Guire <safety@osadl.

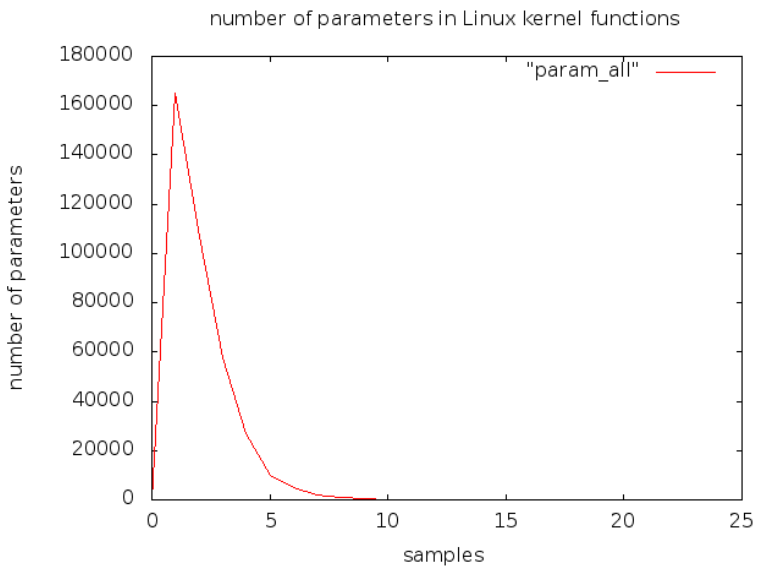
..and resonable number of parameters

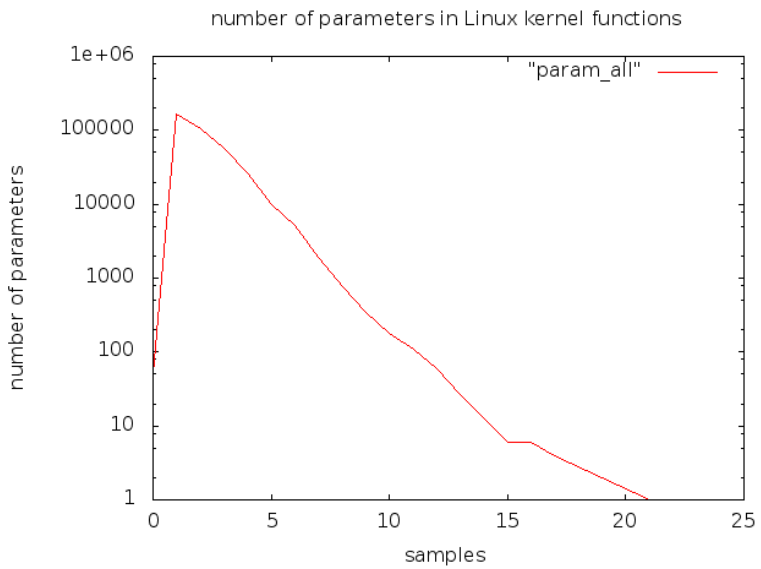
fs/ceph/caps.c:send_cap_msg,line 968 out of control parameter list

```
static int send_cap_msg(struct ceph_mds_session *session,
              u64 ino, u64 cid, int op,
              int caps, int wanted, int dirty,
              u32 seq, u64 flush_tid, u32 issue_seq, u32 mseq,
              u64 size, u64 max_size,
              struct timespec *mtime, struct timespec *atime,
              u64 time_warp_seq,
              kuid_t uid, kgid_t gid, umode_t mode,
              u64 xattr_version,
              struct ceph_buffer *xattrs_buf,
              u64 follows, bool inline_data)
{
```

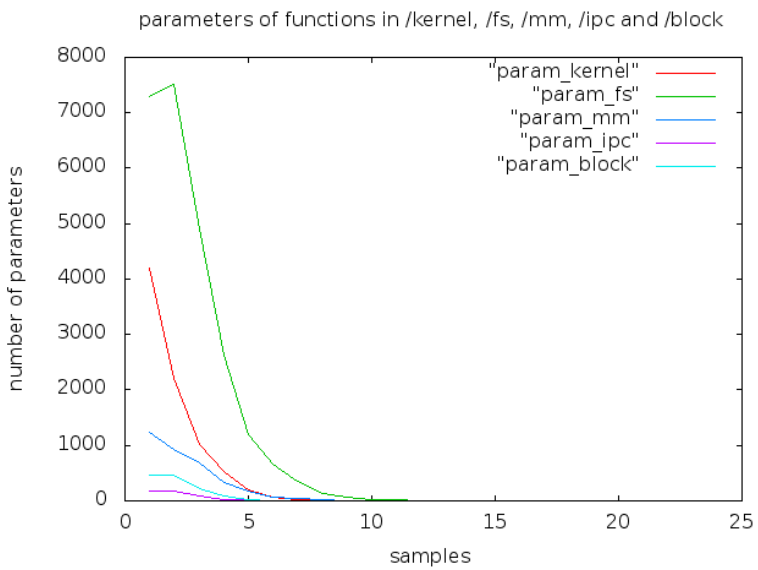
Plain ugly - no excuse for this one - simply exclude ceph from the list of suitable fs.

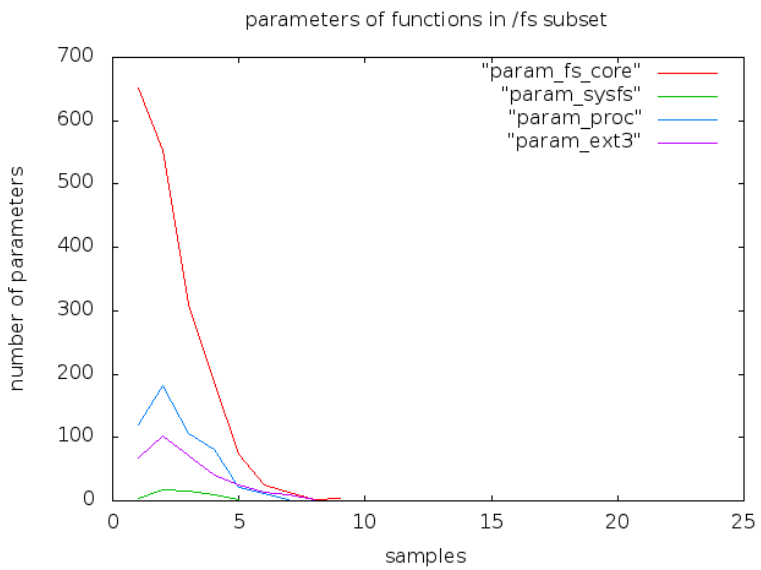
Linux total parameter distribution





Core subset parameter distribution





identifying problem cases

both "bad" functions are in lockdep:

```
<function(name='__lock_acquire',
source_file='kernel/locking/lockdep.c',
line='3068',
column='12',
parameter_number='9')>
<function(name='print_bad_irq_dependency',
source_file='kernel/locking/lockdep.c',
line='1492',
column='1',
parameter_number='10')>
```

Type issues in C

- C is not type safe
- mismatch of types can lead to hard to locate problems
- Automatic type conversion in C hides the problem
- IEC 61508 Ed 2 B.1-8 "Design and coding standards": *No automatic type conversion - highly recomended for SIL2*

Nicholas Mc Guire <safety@osadl.

API compliance - completion

semantic patch	findings	files	confirmed
duplicate_init_completion.cocci	2	2	2
check_for_signal_ignored.cocci	6	4	6
false_declare_completion.cocci	6	5	6
false_init_compltion.cocci	9	6	9
check_unhandled_return.cocci	10	8	4
check_for_negativ_ret.cocci	11	9	3
check_for_return_unused.cocci	62	42	2
check_for_signed_return.cocci	**126**	81	36
check_wrong_context2.cocci	0 (!)	0	-

Type consistency - system components

Component	Nr Functions	Inconsistent	%
kernel	374600	10727	2.85
glibc	9184	268	2.92
busybox	3645	43	1.18

versions: kernel 4.1-rc2, glibc-2.9, busybox-1.2.2.1

Type consistency - kernel core

	kern	mm	ipc	init	net	lib	total	%
wrong	1	1	0	0	1	1	4	0.5
sign	**97**	**65**	**4**	**1**	**218**	**21**	**406**	**47.4**
down sized	**4**	**5**	0	0	**21**	**5**	**35**	**4.0**
up sized	66	34	8	0	123	3	234	27.3
declar ation	8	0	0	0	15	2	25	2.9
false pos	31	17	4	0	89	12	153	17.9
	207	122	16	1	467	44	857	

Type consistency - get_user()

```
arch/alpha/include/asm/uaccess.h:65,get_user() - returns long
arch/arm/include/asm/uaccess.h:199,get_user() - returns int
arch/arm/include/asm/uaccess.h:267,get_user() - returns long
arch/arm64/include/asm/uaccess.h:288,get_user() - returns int
arch/avr32/include/asm/uaccess.h:131,get_user() - returns int
arch/blackfin/include/asm/uaccess.h:129,get_user() - returns int
arch/cris/include/asm/uaccess.h:95,get_user() - returns long
arch/frv/include/asm/uaccess.h:319,get_user() - returns int
arch/ia64/include/asm/uaccess.h:402,get_user() - returns long
arch/m32r/include/asm/uaccess.h:693,get_user() - returns long
arch/m68k/include/asm/uaccess_mm.h:393,get_user() - returns long
arch/m68k/include/asm/uaccess_no.h:181,get_user() - returns int
arch/metag/include/asm/uaccess.h:246,get_user() - returns long
arch/microblaze/include/asm/uaccess.h:426,get_user() - returns int
arch/mips/include/asm/uaccess.h:1445,get_user() - returns int
arch/mn10300/include/asm/uaccess.h:495,get_user() - returns int
arch/nios2/include/asm/uaccess.h:231,get_user() - returns long
arch/openrisc/include/asm/uaccess.h:324,get_user() - returns long
arch/parisc/include/asm/uaccess.h;260,get_user() - returns long
arch/powerpc/include/asm/uaccess.h:454,get_user() - returns long
arch/s390/include/asm/uaccess.h:377,get_user() - returns int
arch/score/include/asm/uaccess.h:424,get_user() - returns long
arch/sh/include/asm/uaccess.h:211,get_user() - returns long
arch/sparc/include/asm/uaccess_32.h:377,get_user() - returns int
arch/sparc/include/asm/uaccess_64.h:289,get_user() - returns int
arch/tile/include/asm/uaccess.h:559,get_user() - returns int
arch/um/include/asm/uaccess.h:178,get_user() - returns int
arch/x86/include/asm/uaccess.h:744,get_user() - returns int
arch/xtensa/include/asm/uaccess.h:510,get_user() - returns long
include/asm-generic/uaccess.h:346,get_user() - returns int
tools/virtio/linux/uaccess.h:50,get_user() - returns int
```

Handling of "bad"-code

Can we handle this ?

- careful selection - review based configuration.
- tools - automate it - formal methods.
- fix those issues in the core code SIL2LinuxMP needs (aprox. 1k patches)
- build up interface to the community - "fix once" is the goal
- push the tools out to the developers (once they are clean)
- build awareness in the community - notably of types

Conclusions

- The code development largely looks stable and can be mapped to SIL2 requirements
- There are some findings that need to be addressed
  - Most can be handled by proper selection
  - Some - notably types - need to be addressed by analysis and cleanup
- There is quite some work to do - no desaster yet
- The kernel as a whole has some QA issues that need to be communicated to the kernel community - and where possible addressed by automated methods.

SIL2LinuxMP will not solve all (not even find all) kernel problems - but we do think we can find -> analyze -> fix issues for the SIL2LinuxMP core and while at it, contribute to improving the general kernel QA.