東京 2025

LINUX PLUMBERS CONFERENCE

TOKYO, JAPAN / DECEMBER 11-13, 2025

Exploring possibilities for integrating StrictDoc and ELISA’s requirements template approach for the Linux kernel

RE: Defining and maintaining requirements in the Linux Kernel

Tobias Deiminger – tobias.deiminger@linutronix.de

東京 2025
LINUX PLUMBERS CONFERENCE
TOKYO, JAPAN / DEC. 11-13, 2025

Agenda

1. Announcement: Showcase on GitHub
2. RFC: Adding a new requirement to Linux
3. RFC: Which items do we want to trace?
4. Q: Refining the Sidecar
5. Q: Integration Sphinx, kernel-doc, StrictDoc
6. RFC: Semantic search for critical functions with Coccinelle

StrictDoc tool

- Created in 2019, inspired by Doorstop
- Apache 2 license, 1.9K pull requests, 5K+ commits, 30K+ LOC
- In a nutshell:
    - **Let's cut prose and code into atomic nodes, give them UUIDs and attributes, and link them together to form a graph**
- Key highlights:
    - Connecting docs, requirements, source and test code, test reports, coverage.
    - Web-based requirements editor.
    - SDoc format for storing requirements with metadata. Internal representation is a graph.
    - Other formats can be read or written. Native ReqIF bi-directional interface.
    - RST export for interfacing with Sphinx. Possible direction: sphinx-strictdoc plugin.
    - Work with the SPDX FuSa WG. Establishing the equivalence between SPDX and SDoc.

RFC: Showcase on GitHub

Linux + ELISA + StrictDoc

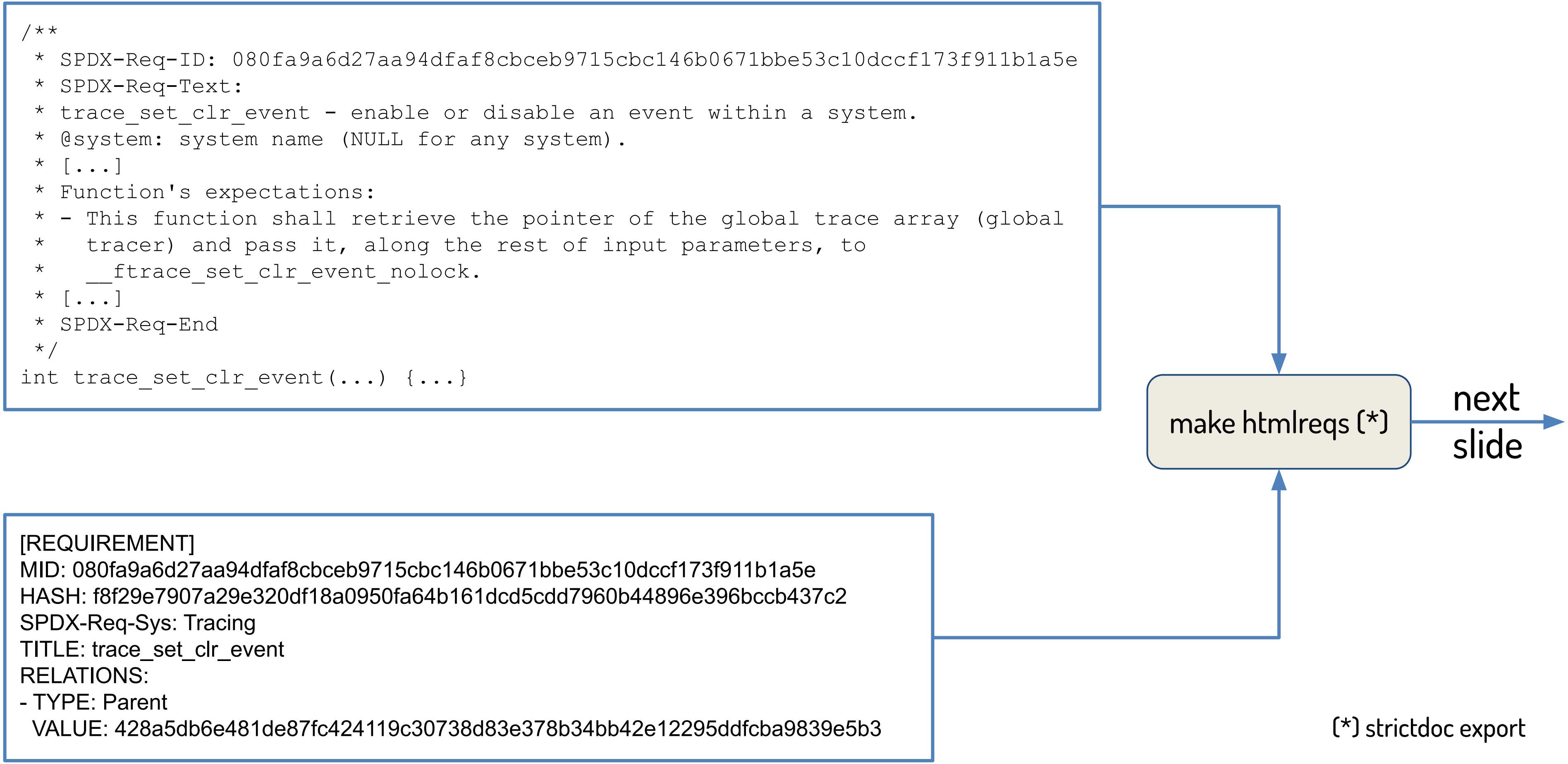
https://github.com/strictdoc-project/linux-strictdoc

Results from CI:

rendered document

traceability graph

Input: Requirements from C Comments and Sidecar

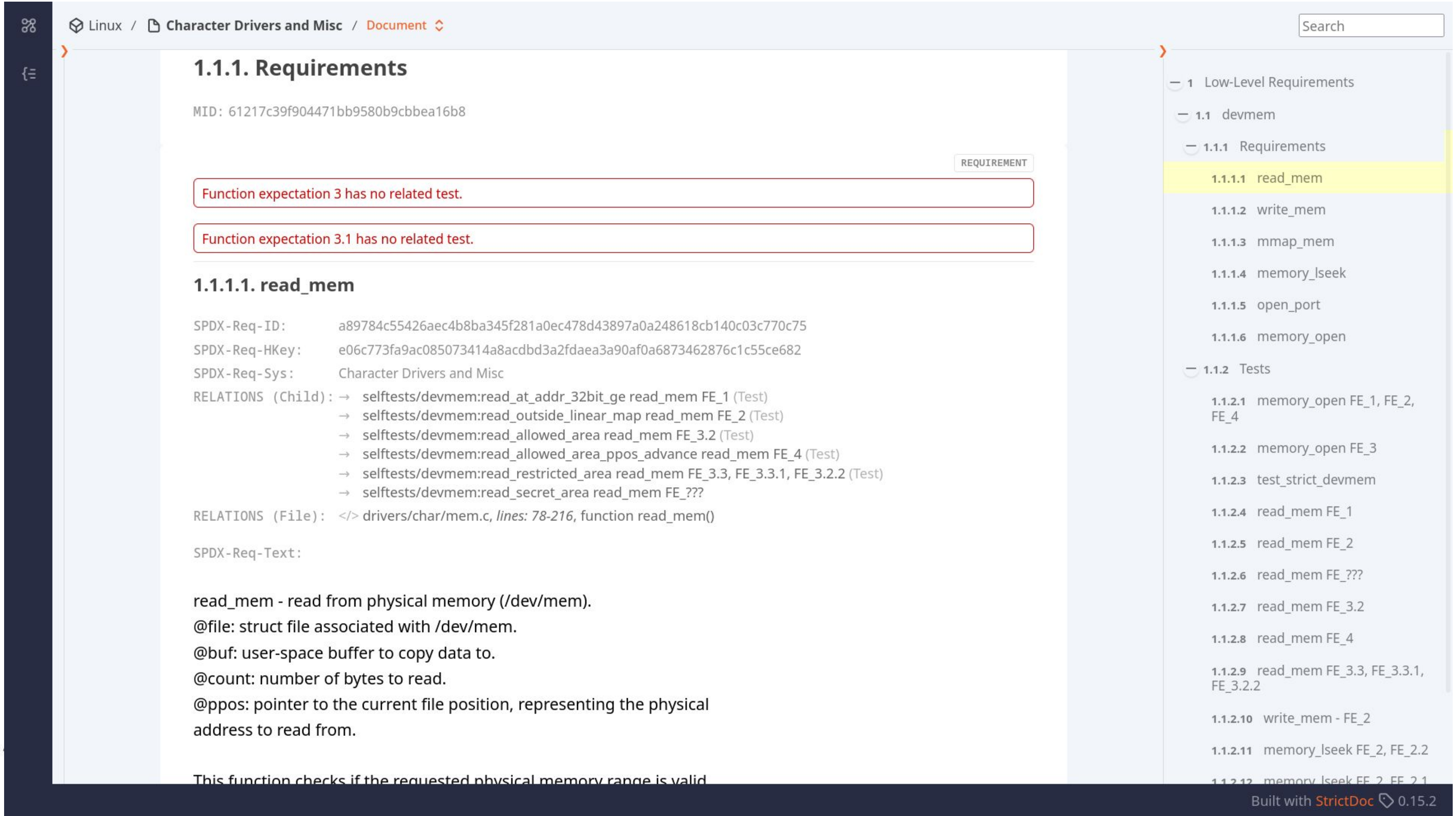


```
/**
 * SPDX-Req-ID: 080fa9a6d27aa94dfaf8cbceb9715cbc146b0671bbe53c10dccf173f911b1a5e
 * SPDX-Req-Text:
 * trace_set_clr_event - enable or disable an event within a system.
 * @system: system name (NULL for any system).
 * [...]
 * Function's expectations:
 * - This function shall retrieve the pointer of the global trace array (global
 *   tracer) and pass it, along the rest of input parameters, to
 *   __ftrace_set_clr_event_nolock.
 * [...]
 * SPDX-Req-End
 */
int trace_set_clr_event(...) {...}
```

```
[REQUIREMENT]
MID: 080fa9a6d27aa94dfaf8cbceb9715cbc146b0671bbe53c10dccf173f911b1a5e
HASH: f8f29e7907a29e320df18a0950fa64b161dcd5cdd7960b44896e396bccb437c2
SPDX-Req-Sys: Tracing
TITLE: trace_set_clr_event
RELATIONS:
- TYPE: Parent
  VALUE: 428a5db6e481de87fc424119c30738d83e378b34bb42e12295ddfcba9839e5b3
```

(*) strictdoc export

Output: Document View with Links and Validations



1.1.1. Requirements

MID: 61217c39f904471bb9580b9cbbea16b8

Function expectation 3 has no related test.

Function expectation 3.1 has no related test.

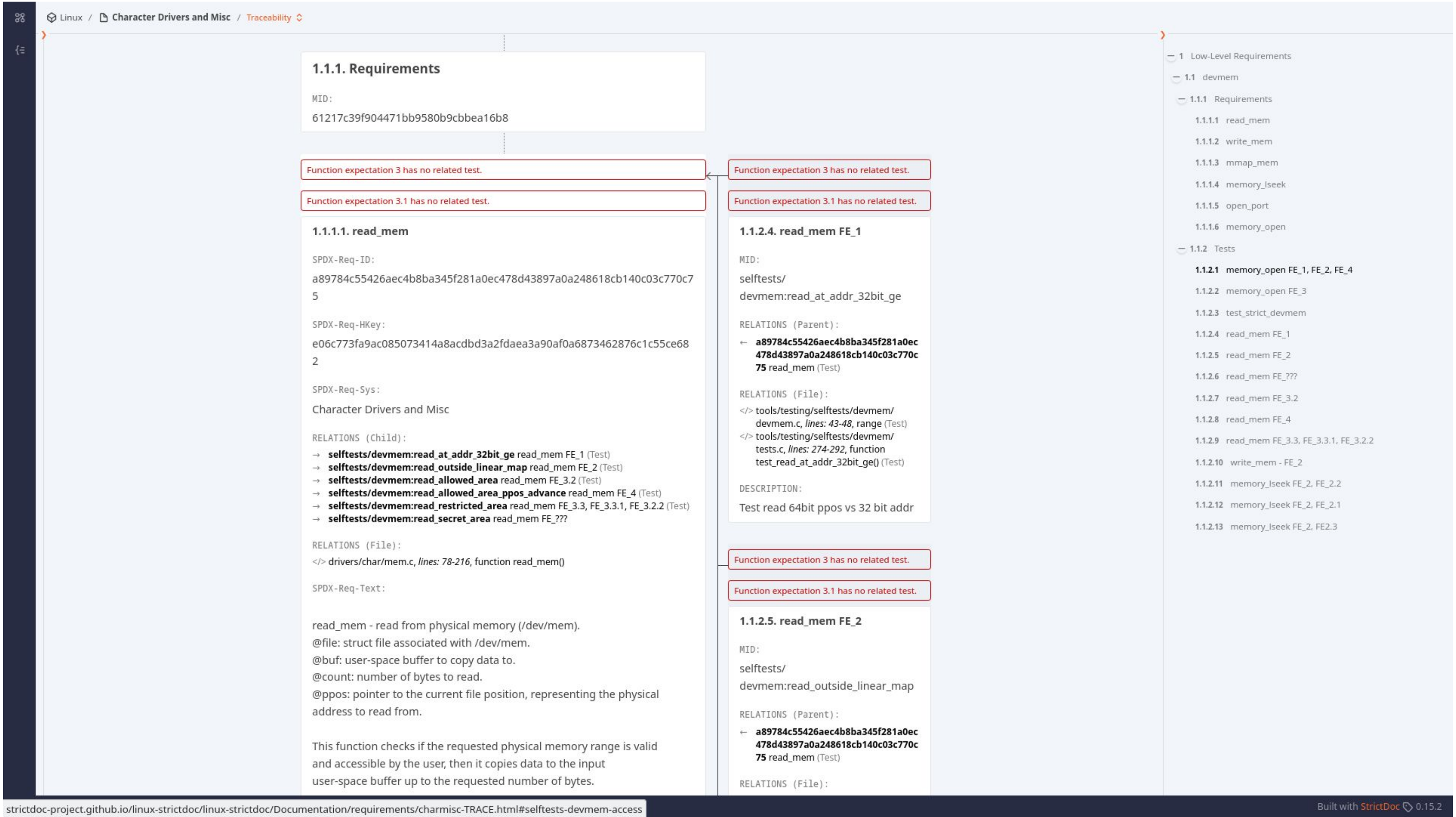
1.1.1.1. read_mem

```
SPDX-Req-ID:          a89784c55426aec4b8ba345f281a0ec478d43897a0a248618cb140c03c770c75
SPDX-Req-HKey:        e06c773fa9ac085073414a8acdbd3a2fdaea3a90af0a6873462876c1c55ce682
SPDX-Req-Sys:         Character Drivers and Misc
RELATIONS (Child):  → selftests/devmem:read_at_addr_32bit_ge read_mem FE_1 (Test)
                    → selftests/devmem:read_outside_linear_map read_mem FE_2 (Test)
                    → selftests/devmem:read_allowed_area read_mem FE_3.2 (Test)
                    → selftests/devmem:read_allowed_area_ppos_advance read_mem FE_4 (Test)
                    → selftests/devmem:read_restricted_area read_mem FE_3.3, FE_3.3.1, FE_3.2.2 (Test)
                    → selftests/devmem:read_secret_area read_mem FE_???
RELATIONS (File):   </> drivers/char/mem.c, lines: 78-216, function read_mem()

SPDX-Req-Text:
```

read_mem - read from physical memory (/dev/mem).
@file: struct file associated with /dev/mem.
@buf: user-space buffer to copy data to.
@count: number of bytes to read.
@ppos: pointer to the current file position, representing the physical address to read from.

This function checks if the requested physical memory range is valid

Output: Traceability View



Linux / Character Drivers and Misc / Traceability

1.1.1. Requirements

MID:

61217c39f904471bb9580b9cbbea16b8

Function expectation 3 has no related test.

Function expectation 3.1 has no related test.

1.1.1.1. read_mem

SPDX-Req-ID:

a89784c55426aec4b8ba345f281a0ec478d43897a0a248618cb140c03c770c75

SPDX-Req-HKey:

e06c773fa9ac085073414a8acdbd3a2fdaea3a90af0a6873462876c1c55ce682

SPDX-Req-Sys:

Character Drivers and Misc

RELATIONS (Child):

→ **selftests/devmem:read_at_addr_32bit_ge** read_mem FE_1 (Test)
→ **selftests/devmem:read_outside_linear_map** read_mem FE_2 (Test)
→ **selftests/devmem:read_allowed_area** read_mem FE_3.2 (Test)
→ **selftests/devmem:read_allowed_area_ppos_advance** read_mem FE_4 (Test)
→ **selftests/devmem:read_restricted_area** read_mem FE_3.3, FE_3.3.1, FE_3.2.2 (Test)
→ **selftests/devmem:read_secret_area** read_mem FE_???

RELATIONS (File):

</> drivers/char/mem.c, *lines: 78-216*, function read_mem()

SPDX-Req-Text:

read_mem - read from physical memory (/dev/mem).
@file: struct file associated with /dev/mem.
@buf: user-space buffer to copy data to.
@count: number of bytes to read.
@ppos: pointer to the current file position, representing the physical address to read from.

This function checks if the requested physical memory range is valid and accessible by the user, then it copies data to the input user-space buffer up to the requested number of bytes.

Function expectation 3 has no related test.

Function expectation 3.1 has no related test.

1.1.2.4. read_mem FE_1

MID:

selftests/devmem:read_at_addr_32bit_ge

RELATIONS (Parent):

← **a89784c55426aec4b8ba345f281a0ec478d43897a0a248618cb140c03c770c75** read_mem (Test)

RELATIONS (File):

</> tools/testing/selftests/devmem/devmem.c, *lines: 43-48*, range (Test)
</> tools/testing/selftests/devmem/tests.c, *lines: 274-292*, function test_read_at_addr_32bit_ge() (Test)

DESCRIPTION:

Test read 64bit ppos vs 32 bit addr

Function expectation 3 has no related test.

Function expectation 3.1 has no related test.

1.1.2.5. read_mem FE_2

MID:

selftests/devmem:read_outside_linear_map

RELATIONS (Parent):

← **a89784c55426aec4b8ba345f281a0ec478d43897a0a248618cb140c03c770c75** read_mem (Test)

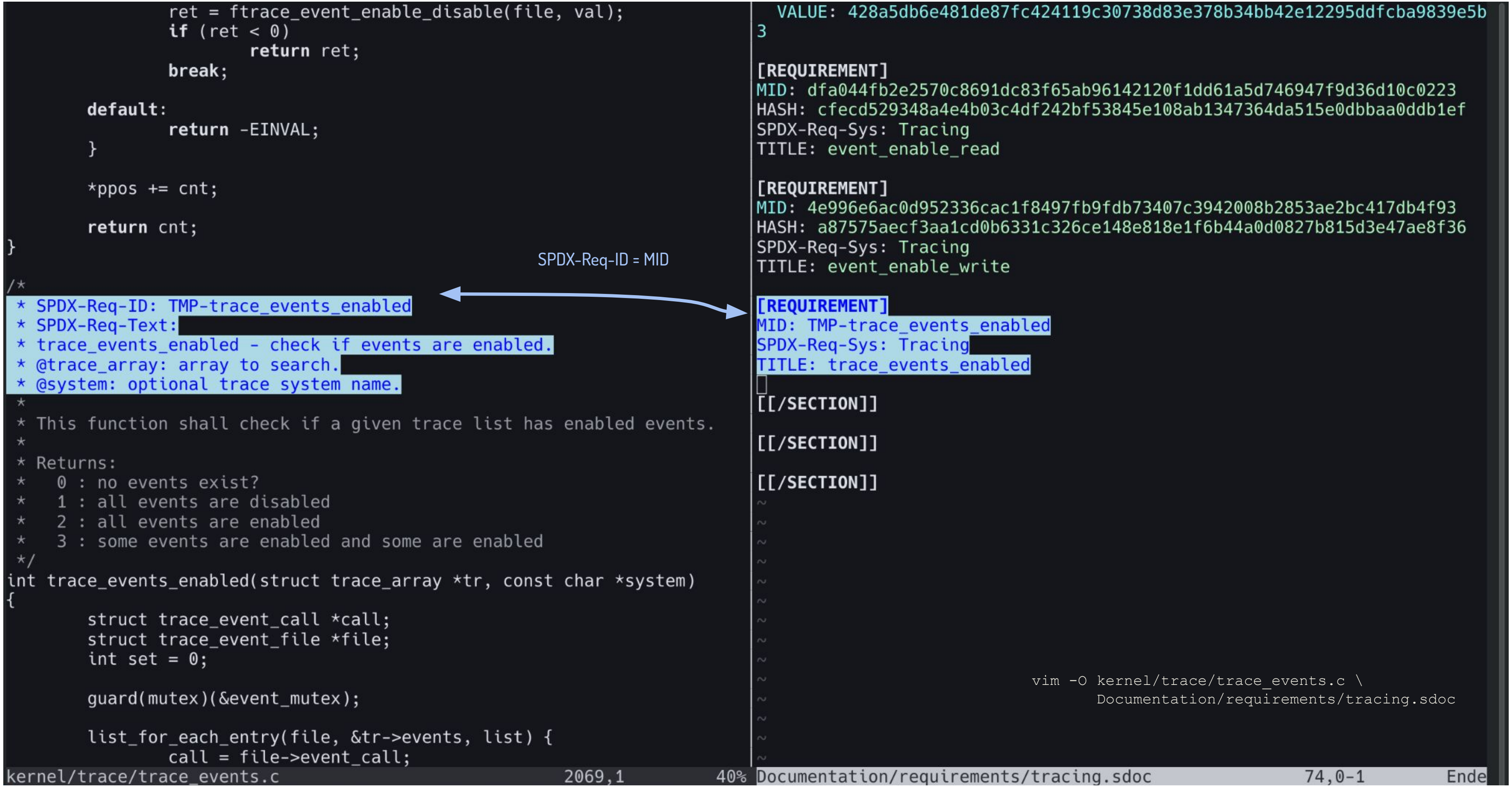
RELATIONS (File):

strictdoc-project.github.io/linux-strictdoc/linux-strictdoc/Documentation/requirements/charmisc-TRACE.html#selftests-devmem-access

Built with StrictDoc 0.15.2

Q/A: Adding a new requirement to Linux

```
pipx install strictdoc
```



```c
                ret = ftrace_event_enable_disable(file, val);
                if (ret < 0)
                        return ret;
                break;

        default:
                return -EINVAL;
        }

        *ppos += cnt;

        return cnt;
}

/*
 * SPDX-Req-ID: TMP-trace_events_enabled
 * SPDX-Req-Text:
 * trace_events_enabled - check if events are enabled.
 * @trace_array: array to search.
 * @system: optional trace system name.
 *
 * This function shall check if a given trace list has enabled events.
 *
 * Returns:
 *   0 : no events exist?
 *   1 : all events are disabled
 *   2 : all events are enabled
 *   3 : some events are enabled and some are enabled
 */
int trace_events_enabled(struct trace_array *tr, const char *system)
{
        struct trace_event_call *call;
        struct trace_event_file *file;
        int set = 0;

        guard(mutex)(&event_mutex);

        list_for_each_entry(file, &tr->events, list) {
                call = file->event_call;
kernel/trace/trace_events.c                          2069,1         40%
```

SPDX-Req-ID = MID

```
  VALUE: 428a5db6e481de87fc424119c30738d83e378b34bb42e12295ddfcba9839e5b
3

[REQUIREMENT]
MID: dfa044fb2e2570c8691dc83f65ab96142120f1dd61a5d746947f9d36d10c0223
HASH: cfecd529348a4e4b03c4df242bf53845e108ab1347364da515e0dbbaa0ddb1ef
SPDX-Req-Sys: Tracing
TITLE: event_enable_read

[REQUIREMENT]
MID: 4e996e6ac0d952336cac1f8497fb9fdb73407c3942008b2853ae2bc417db4f93
HASH: a87575aecf3aa1cd0b6331c326ce148e818e1f6b44a0d0827b815d3e47ae8f36
SPDX-Req-Sys: Tracing
TITLE: event_enable_write

[REQUIREMENT]
MID: TMP-trace_events_enabled
SPDX-Req-Sys: Tracing
TITLE: trace_events_enabled

[[/SECTION]]

[[/SECTION]]

[[/SECTION]]
~
~
~
~
~
~
~
~
~
~
~
~
~
~
Documentation/requirements/tracing.sdoc          74,0-1         Ende
```

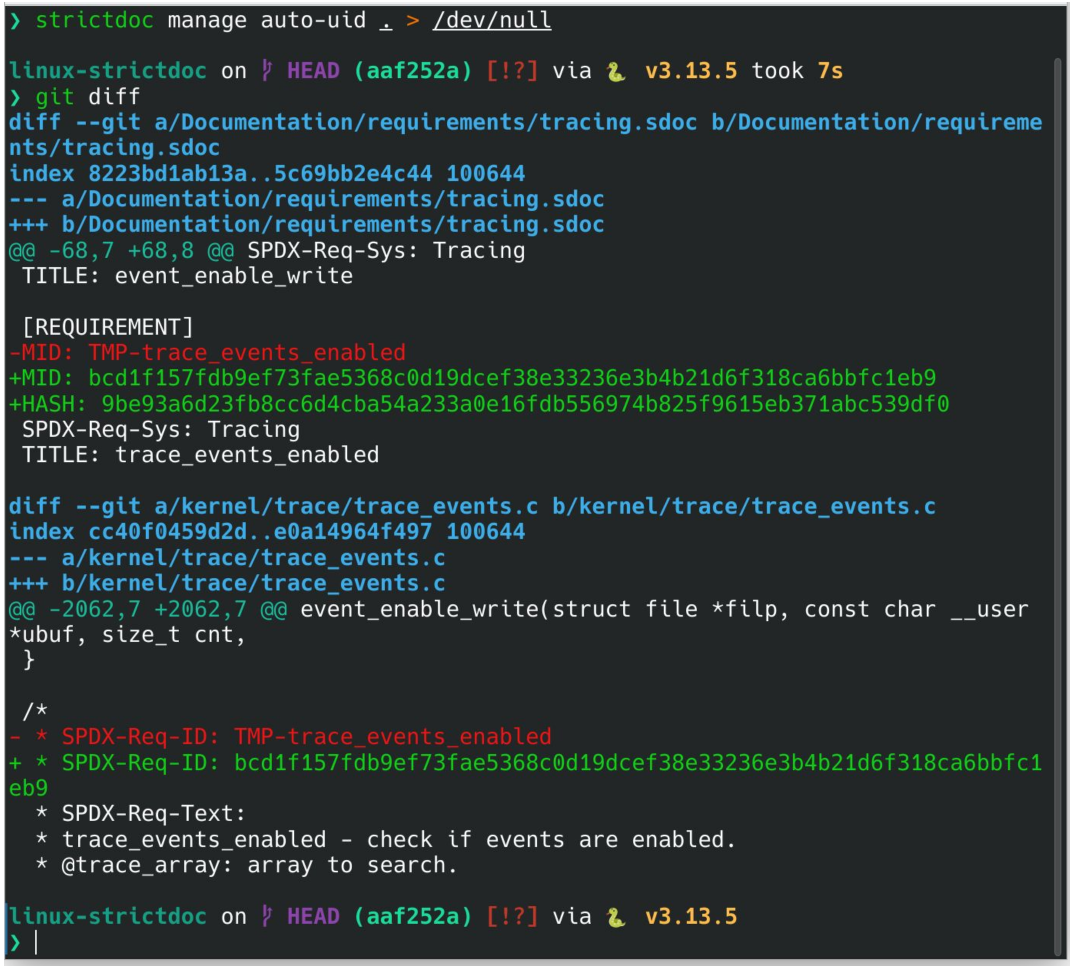
```
vim -O kernel/trace/trace_events.c \
       Documentation/requirements/tracing.sdoc
```

Q/A: Adding a new requirement to Linux

Create IDs

```
strictdoc manage auto-uid .
```

generated MID
generated HASH

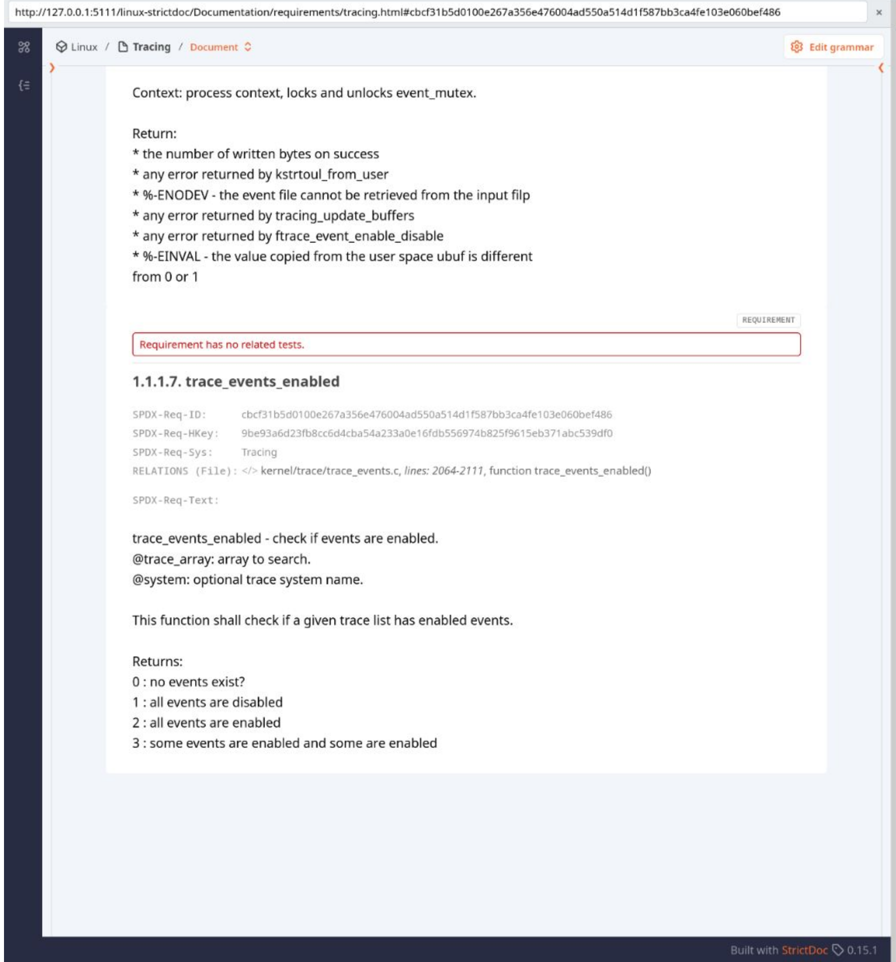


```
❯ strictdoc manage auto-uid . > /dev/null

linux-strictdoc on  HEAD (aaf252a) [!?] via 🐍 v3.13.5 took 7s
❯ git diff
diff --git a/Documentation/requirements/tracing.sdoc b/Documentation/requireme
nts/tracing.sdoc
index 8223bd1ab13a..5c69bb2e4c44 100644
--- a/Documentation/requirements/tracing.sdoc
+++ b/Documentation/requirements/tracing.sdoc
@@ -68,7 +68,8 @@ SPDX-Req-Sys: Tracing
 TITLE: event_enable_write

 [REQUIREMENT]
-MID: TMP-trace_events_enabled
+MID: bcd1f157fdb9ef73fae5368c0d19dcef38e33236e3b4b21d6f318ca6bbfc1eb9
+HASH: 9be93a6d23fb8cc6d4cba54a233a0e16fdb556974b825f9615eb371abc539df0
 SPDX-Req-Sys: Tracing
 TITLE: trace_events_enabled

diff --git a/kernel/trace/trace_events.c b/kernel/trace/trace_events.c
index cc40f0459d2d..e0a14964f497 100644
--- a/kernel/trace/trace_events.c
+++ b/kernel/trace/trace_events.c
@@ -2062,7 +2062,7 @@ event_enable_write(struct file *filp, const char __user
*ubuf, size_t cnt,
 }

 /*
- * SPDX-Req-ID: TMP-trace_events_enabled
+ * SPDX-Req-ID: bcd1f157fdb9ef73fae5368c0d19dcef38e33236e3b4b21d6f318ca6bbfc1
eb9
  * SPDX-Req-Text:
  * trace_events_enabled - check if events are enabled.
  * @trace_array: array to search.

linux-strictdoc on  HEAD (aaf252a) [!?] via 🐍 v3.13.5
❯ |
```

東京2025
LINUX
PLUMBERS CONFERENCE
TOKYO, JAPAN / DEC. 11-13, 2025

Q/A: Adding a new requirement to Linux: **Compile**

`strictdoc export .`



http://127.0.0.1:5111/linux-strictdoc/Documentation/requirements/tracing.html#cbcf31b5d0100e267a356e476004ad550a514d1f587bb3ca4fe103e060bef486

Context: process context, locks and unlocks event_mutex.

Return:
* the number of written bytes on success
* any error returned by kstrtoul_from_user
* %-ENODEV - the event file cannot be retrieved from the input filp
* any error returned by tracing_update_buffers
* any error returned by ftrace_event_enable_disable
* %-EINVAL - the value copied from the user space ubuf is different from 0 or 1

Requirement has no related tests.

1.1.1.7. trace_events_enabled

SPDX-Req-ID: cbcf31b5d0100e267a356e476004ad550a514d1f587bb3ca4fe103e060bef486
SPDX-Req-HKey: 9be93a6d23fb8cc6d4cba54a233a0e16fdb556974b825f9615eb371abc539df0
SPDX-Req-Sys: Tracing
RELATIONS (File): </> kernel/trace/trace_events.c, *lines: 2064-2111*, function trace_events_enabled()

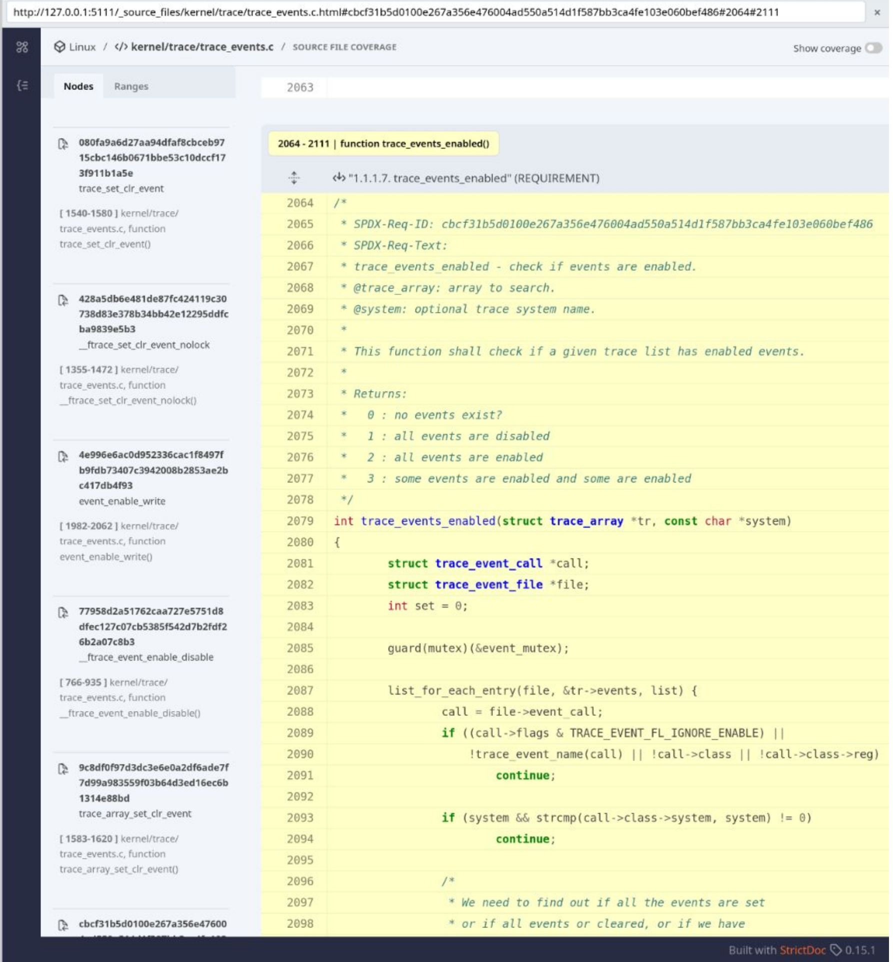
SPDX-Req-Text:

trace_events_enabled - check if events are enabled.
@trace_array: array to search.
@system: optional trace system name.

This function shall check if a given trace list has enabled events.

Returns:
0 : no events exist?
1 : all events are disabled
2 : all events are enabled
3 : some events are enabled and some are enabled

Built with StrictDoc 0.15.1



http://127.0.0.1:5111/_source_files/kernel/trace/trace_events.c.html#cbcf31b5d0100e267a356e476004ad550a514d1f587bb3ca4fe103e060bef486#2064#2111

Linux / </> kernel/trace/trace_events.c / SOURCE FILE COVERAGE
Show coverage
Nodes
Ranges

080fa9a6d27aa94dfaf8cbceb9715cbc146b0671bbe53c10dccf173f911b1a5e
trace_set_clr_event
[1540-1580] kernel/trace/trace_events.c, function trace_set_clr_event()

428a5db6e481de87fc424119c30738d83e378b34bb42e12295ddfcba9839e5b3
__ftrace_set_clr_event_nolock
[1355-1472] kernel/trace/trace_events.c, function __ftrace_set_clr_event_nolock()

4e996e6ac0d952336cac1f8497fb9fdb73407c3942008b2853ae2bc417db4f93
event_enable_write
[1982-2062] kernel/trace/trace_events.c, function event_enable_write()

77958d2a51762caa727e5751d8dfec127c07cb5385f542d7b2fdf26b2a07c8b3
__ftrace_event_enable_disable
[766-935] kernel/trace/trace_events.c, function __ftrace_event_enable_disable()

9c8df0f97d3dc3e6e0a2df6ade7f7d99a983559f03b64d3ed16ec6b1314e88bd
trace_array_set_clr_event
[1583-1620] kernel/trace/trace_events.c, function trace_array_set_clr_event()

cbcf31b5d0100e267a356e47600

2064 - 2111 | function trace_events_enabled()

"1.1.1.7. trace_events_enabled" (REQUIREMENT)

```c
/*
 * SPDX-Req-ID: cbcf31b5d0100e267a356e476004ad550a514d1f587bb3ca4fe103e060bef486
 * SPDX-Req-Text:
 * trace_events_enabled - check if events are enabled.
 * @trace_array: array to search.
 * @system: optional trace system name.
 *
 * This function shall check if a given trace list has enabled events.
 *
 * Returns:
 *   0 : no events exist?
 *   1 : all events are disabled
 *   2 : all events are enabled
 *   3 : some events are enabled and some are enabled
 */
int trace_events_enabled(struct trace_array *tr, const char *system)
{
        struct trace_event_call *call;
        struct trace_event_file *file;
        int set = 0;

        guard(mutex)(&event_mutex);

        list_for_each_entry(file, &tr->events, list) {
                call = file->event_call;
                if ((call->flags & TRACE_EVENT_FL_IGNORE_ENABLE) ||
                    !trace_event_name(call) || !call->class || !call->class->reg)
                        continue;

                if (system && strcmp(call->class->system, system) != 0)
                        continue;

                /*
                 * We need to find out if all the events are set
                 * or if all events or cleared, or if we have
```

Q/A: Adding a new requirement to Linux

Submit for Review

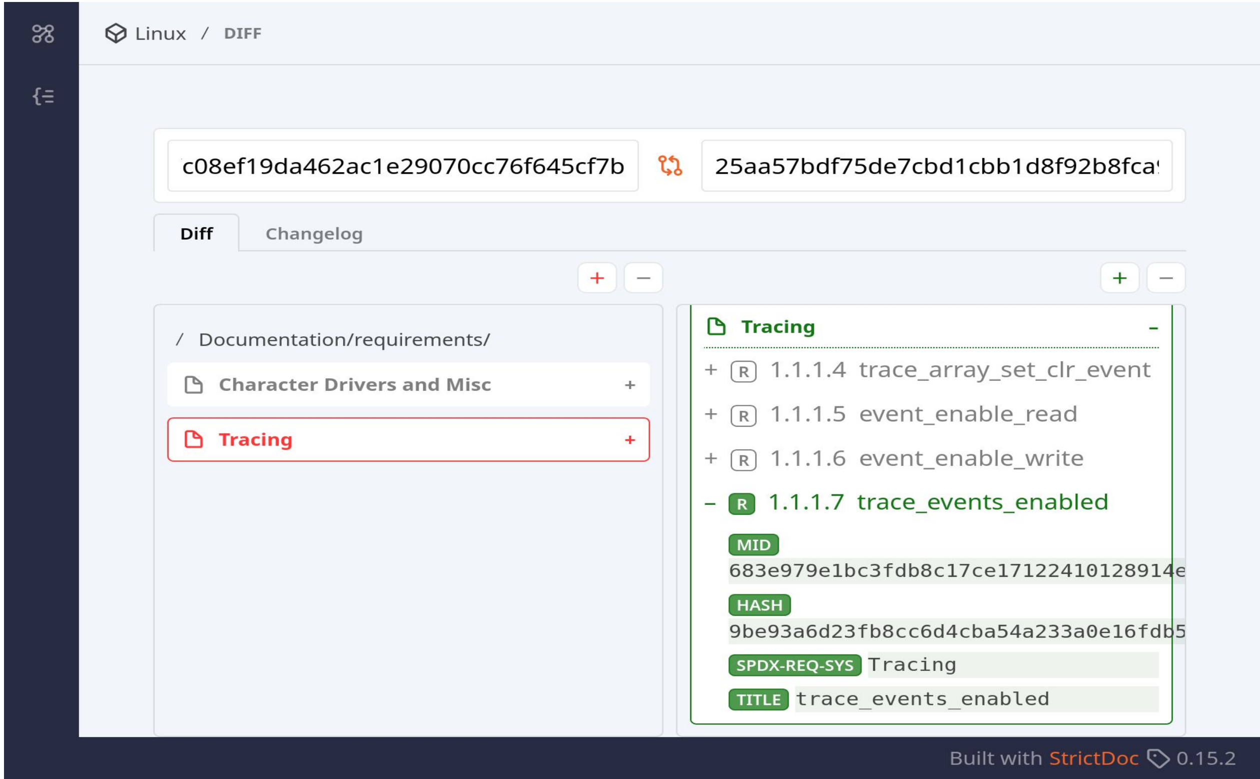
Developer → Maintainer

Developer:

```
git format-patch
git send-email
```

Maintainer:

```
strictdoc manage auto-uid . && git diff
strictdoc export --generate-diff-git="HEAD^..HEAD"
```



Linux / DIFF

c08ef19da462ac1e29070cc76f645cf7b ⇄ 25aa57bdf75de7cbd1cbb1d8f92b8fca

Diff | Changelog

/ Documentation/requirements/

- Character Drivers and Misc +
- Tracing +

Tracing –

- + R 1.1.1.4 trace_array_set_clr_event
- + R 1.1.1.5 event_enable_read
- + R 1.1.1.6 event_enable_write
- – R 1.1.1.7 trace_events_enabled

MID 683e979e1bc3fdb8c17ce17122410128914e

HASH 9be93a6d23fb8cc6d4cba54a233a0e16fdb5

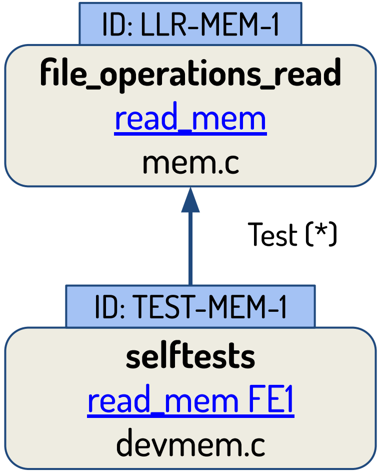
SPDX-REQ-SYS Tracing

TITLE trace_events_enabled

Built with StrictDoc 0.15.2

RFC: Which Items do we Want to Trace?

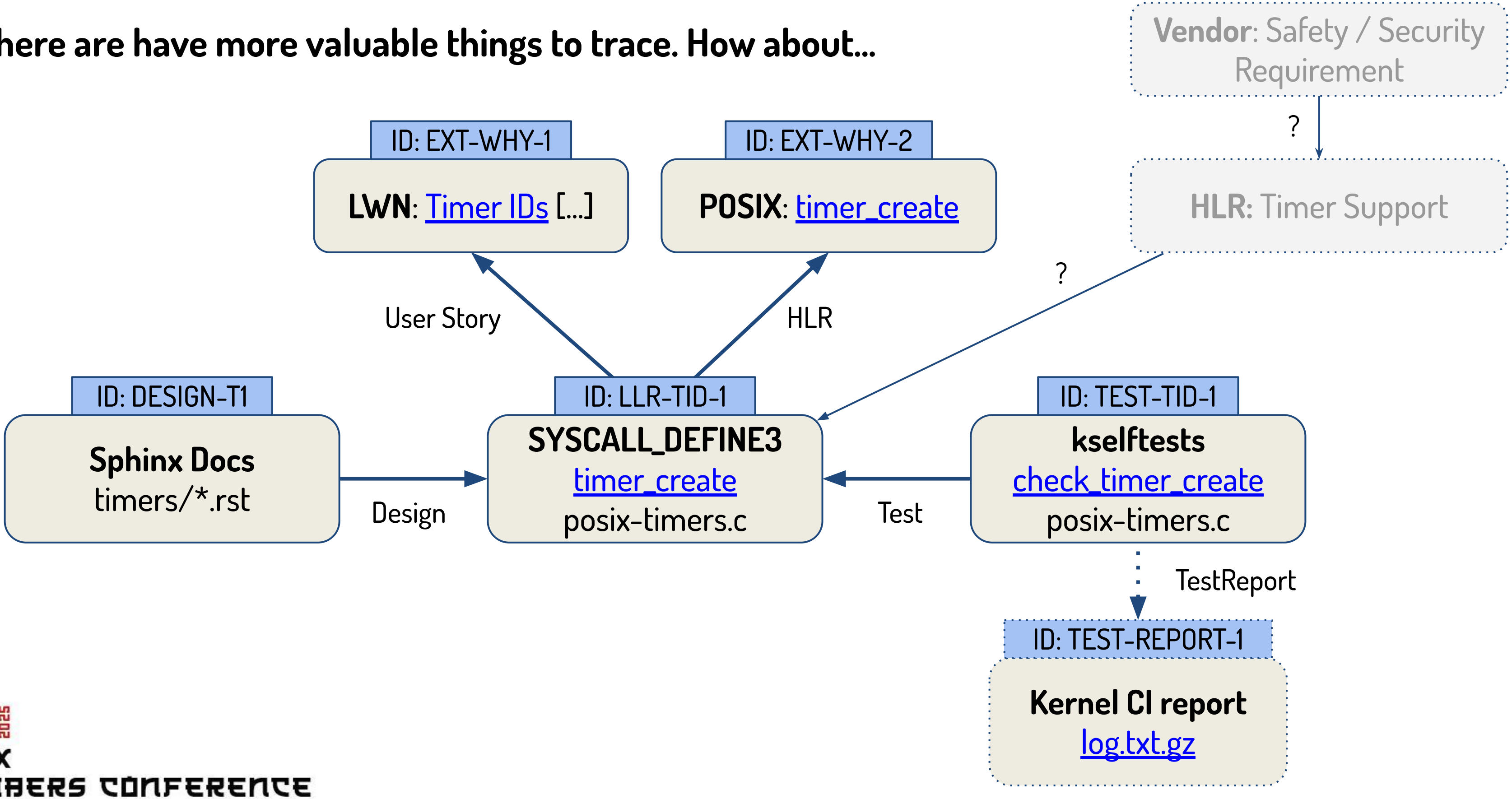
Current Demo: Low Level Requirements and Tests



(*) relation by sidecar shown in appendix, to be reconsidered

RFC: Which items do we want to trace?

There are have more valuable things to trace. How about...



東京 2025 LINUX PLUMBERS CONFERENCE
TOKYO, JAPAN / DEC. 11-13, 2025

Q: Refining the Sidecar

- Sidecar Pro:
  - Keeps Meta-Data out of Code
  - The place where typical StrictDoc projects have their requirements, not only meta data
  - Obvious place for HLRs

Hint: Sidecar = all *.sdoc files in the demo

Q: Refining the Sidecar - To be clarified

- **Structure wanted:** How do .c and .sdoc correlate? One .sdoc per one .c file? One sdoc per subsystem?

  Mirroring structure creates maintenance overhead.

- **Home wanted:** Sidecar files probably committed to Linux tree, but could also be a separate project

- **Format wanted:**

  Should be a “standard” format

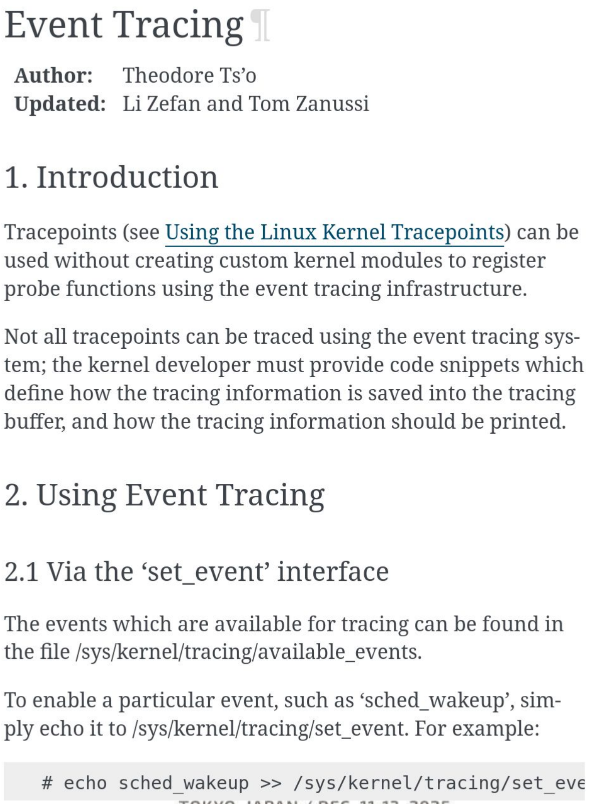
  - sdoc to become an SPDX serialization format?

- Should be human- and machine readable.
  - JSON(-LD) not human readable. sdoc is great in that regard.

Q: Integration Sphinx, kernel-doc, StrictDoc

```
make htmldocs
```

```
make htmlreqs
```



Event Tracing ¶

Author: Theodore Ts'o
Updated: Li Zefan and Tom Zanussi

1. Introduction

Tracepoints (see Using the Linux Kernel Tracepoints) can be used without creating custom kernel modules to register probe functions using the event tracing infrastructure.

Not all tracepoints can be traced using the event tracing system; the kernel developer must provide code snippets which define how the tracing information is saved into the tracing buffer, and how the tracing information should be printed.

2. Using Event Tracing

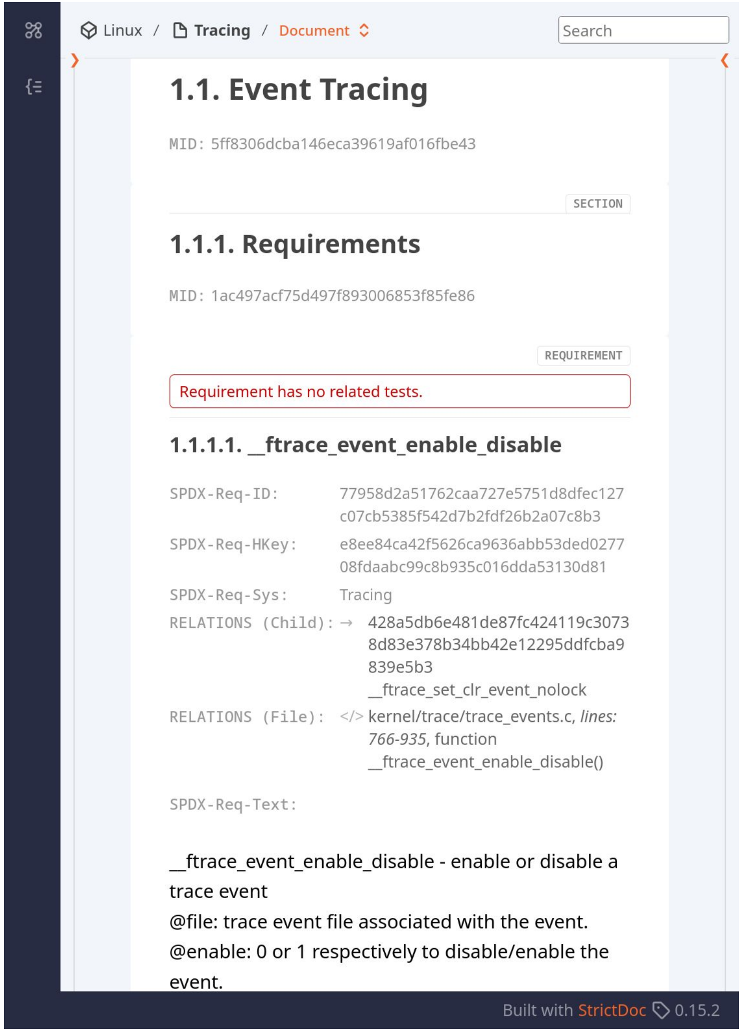
2.1 Via the 'set_event' interface

The events which are available for tracing can be found in the file /sys/kernel/tracing/available_events.

To enable a particular event, such as 'sched_wakeup', simply echo it to /sys/kernel/tracing/set_event. For example:

```
# echo sched_wakeup >> /sys/kernel/tracing/set_eve
```

?



1.1. Event Tracing

MID: 5ff8306dcba146eca39619af016fbe43

SECTION

1.1.1. Requirements

MID: 1ac497acf75d497f893006853f85fe86

REQUIREMENT

Requirement has no related tests.

1.1.1.1. __ftrace_event_enable_disable

SPDX-Req-ID:	77958d2a51762caa727e5751d8dfec127c07cb5385f542d7b2fdf26b2a07c8b3
SPDX-Req-HKey:	e8ee84ca42f5626ca9636abb53ded027708fdaabc99c8b935c016dda53130d81
SPDX-Req-Sys:	Tracing
RELATIONS (Child):	→ 428a5db6e481de87fc424119c30738d83e378b34bb42e12295ddfcba9839e5b3 __ftrace_set_clr_event_nolock
RELATIONS (File):	</> kernel/trace/trace_events.c, *lines: 766-935*, function __ftrace_event_enable_disable()

SPDX-Req-Text:

__ftrace_event_enable_disable - enable or disable a trace event
@file: trace event file associated with the event.
@enable: 0 or 1 respectively to disable/enable the event.

Built with StrictDoc 0.15.2

RFC: Semantic search for critical functions with Coccinelle

How to spot the **most critical functions**? (*)

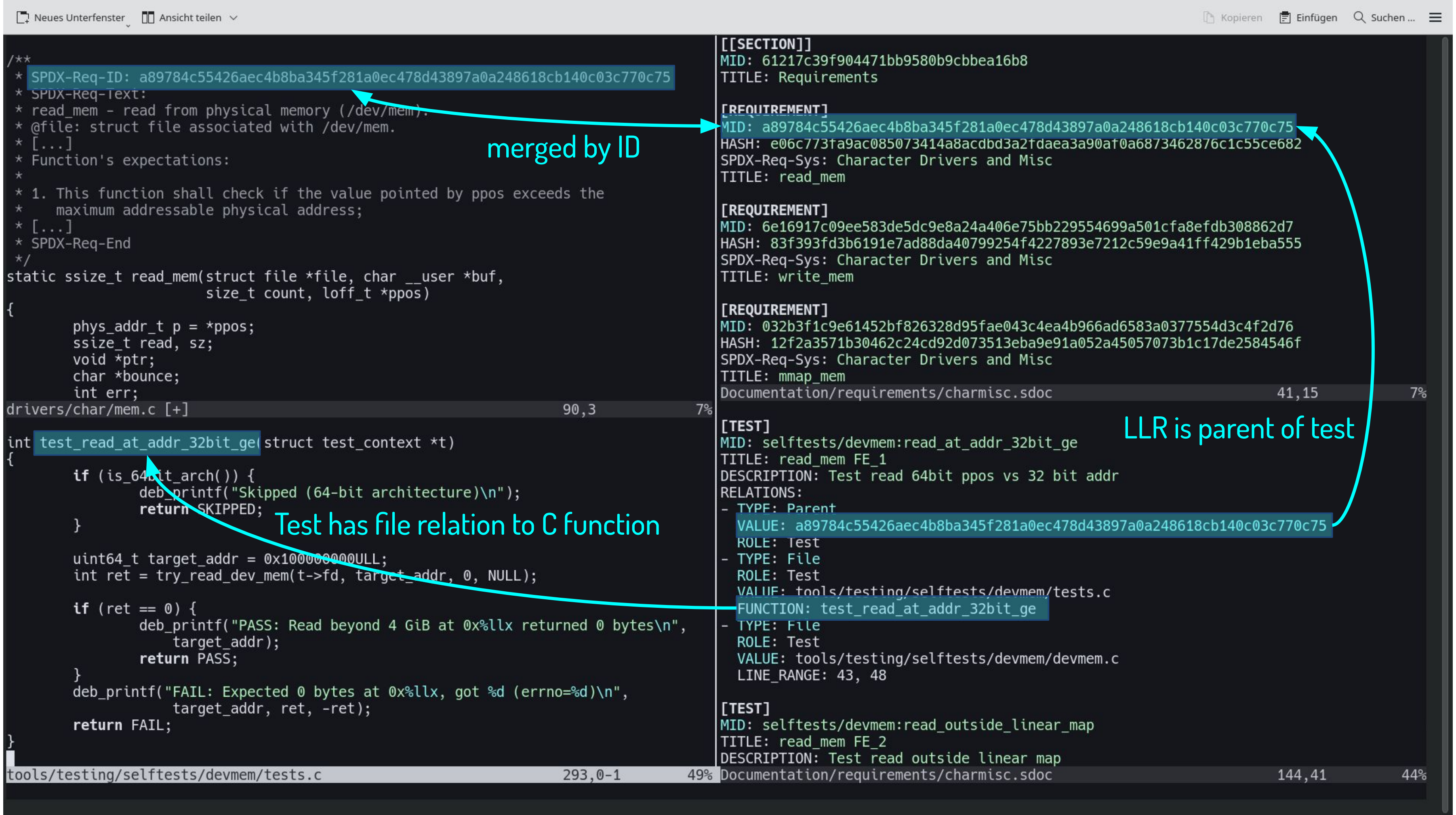
How can developers check if they missed something important?

It's reasonable to consider **internal and external API** as critical. The ELISA paper names them, and the GitHub demo adds a **SmPL** search for the following:

- Syscalls
- Sysfs attributes
- Exported symbols
- File-like operations

(*) naive assumption: maintainers, developers and industry agree on what is most critical

Appendix: Screenshot: **Trace to Tests**



```
/**
 * SPDX-Req-ID: a89784c55426aec4b8ba345f281a0ec478d43897a0a248618cb140c03c770c75
 * SPDX-Req-Text:
 * read_mem - read from physical memory (/dev/mem).
 * @file: struct file associated with /dev/mem.
 * [...]
 * Function's expectations:
 *
 * 1. This function shall check if the value pointed by ppos exceeds the
 *    maximum addressable physical address;
 * [...]
 * SPDX-Req-End
 */
static ssize_t read_mem(struct file *file, char __user *buf,
                        size_t count, loff_t *ppos)
{
        phys_addr_t p = *ppos;
        ssize_t read, sz;
        void *ptr;
        char *bounce;
        int err;
drivers/char/mem.c [+]                                          90,3           7%
```

```
int test_read_at_addr_32bit_ge(struct test_context *t)
{
        if (is_64bit_arch()) {
                deb_printf("Skipped (64-bit architecture)\n");
                return SKIPPED;
        }

        uint64_t target_addr = 0x100000000ULL;
        int ret = try_read_dev_mem(t->fd, target_addr, 0, NULL);

        if (ret == 0) {
                deb_printf("PASS: Read beyond 4 GiB at 0x%llx returned 0 bytes\n",
                           target_addr);
                return PASS;
        }
        deb_printf("FAIL: Expected 0 bytes at 0x%llx, got %d (errno=%d)\n",
                   target_addr, ret, -ret);
        return FAIL;
}

tools/testing/selftests/devmem/tests.c                          293,0-1       49%
```

```
[[SECTION]]
MID: 61217c39f904471bb9580b9cbbea16b8
TITLE: Requirements

[REQUIREMENT]
MID: a89784c55426aec4b8ba345f281a0ec478d43897a0a248618cb140c03c770c75
HASH: e06c773fa9ac085073414a8acdbd3a2fdaea3a90af0a6873462876c1c55ce682
SPDX-Req-Sys: Character Drivers and Misc
TITLE: read_mem

[REQUIREMENT]
MID: 6e16917c09ee583de5dc9e8a24a406e75bb229554699a501cfa8efdb308862d7
HASH: 83f393fd3b6191e7ad88da40799254f4227893e7212c59e9a41ff429b1eba555
SPDX-Req-Sys: Character Drivers and Misc
TITLE: write_mem

[REQUIREMENT]
MID: 032b3f1c9e61452bf826328d95fae043c4ea4b966ad6583a0377554d3c4f2d76
HASH: 12f2a3571b30462c24cd92d073513eba9e91a052a45057073b1c17de2584546f
SPDX-Req-Sys: Character Drivers and Misc
TITLE: mmap_mem
Documentation/requirements/charmisc.sdoc                        41,15          7%
```

```
[TEST]
MID: selftests/devmem:read_at_addr_32bit_ge
TITLE: read_mem FE_1
DESCRIPTION: Test read 64bit ppos vs 32 bit addr
RELATIONS:
- TYPE: Parent
  VALUE: a89784c55426aec4b8ba345f281a0ec478d43897a0a248618cb140c03c770c75
  ROLE: Test
- TYPE: File
  ROLE: Test
  VALUE: tools/testing/selftests/devmem/tests.c
  FUNCTION: test_read_at_addr_32bit_ge
- TYPE: File
  ROLE: Test
  VALUE: tools/testing/selftests/devmem/devmem.c
  LINE_RANGE: 43, 48

[TEST]
MID: selftests/devmem:read_outside_linear_map
TITLE: read_mem FE_2
DESCRIPTION: Test read outside linear map
Documentation/requirements/charmisc.sdoc                        144,41        44%
```

merged by ID

LLR is parent of test

Test has file relation to C function

Appendix: Listing: Find all Exported Symbols with Coccinelle

```
// Exported Symbols

@export_symbol@
declarer name EXPORT_SYMBOL;
declarer name EXPORT_SYMBOL_GPL;
identifier fn;
@@
(
  EXPORT_SYMBOL(fn);
|
  EXPORT_SYMBOL_GPL(fn);
)

@export_symbol_fn@
identifier export_symbol.fn;
position p_fn;
@@
  fn@p_fn(...) {...}

@script:python@
fn << export_symbol.fn;
p << export_symbol_fn.p_fn;
@@
print(f"exported function: {fn} at {p[0].file}:{p[0].line}")
```

Appendix: Listing: **Result of Coccinelle Run**

```
syscall: alpha_pipe in ./arch/alpha/kernel/osf_sys.c:1300
syscall: getdtablesize in ./arch/alpha/kernel/osf_sys.c:553
...
exported function: marvel_ioportmap at ./arch/alpha/kernel/core_marvel.c:797
exported function: marvel_ioread8 at ./arch/alpha/kernel/core_marvel.c:804
...
mmap_f (kvm_gmem_fops): kvm_gmem_mmap at ./virt/kvm/guest_memfd.c:397
read (sof_msg_inject_fops): sof_msg_inject_dfs_read at
./sound/soc/sof/sof-client-ipc-msg-injector.c:52
..
sysfs show (RO): device_show at ./arch/arm/mach-rpc/ecard.c:790
sysfs show (RO): dma_show at ./arch/arm/mach-rpc/ecard.c:760
...
Found 41441 elements
```

LPC 2025 - Overview

Conference Details

The Linux Plumbers Conference is the premier event for developers working at all levels of the plumbing layer and beyond.

Taking place on Thursday 11th, Friday 12th and Saturday 13th of December, this year we will be both in person and remote (hybrid). However to minimize technical issues, we'd appreciate most of the content presenters being in-person.

The in-person venue is the Toranomon Hills Forum, Tokyo, Japan

Toranomon Hills Mori Tower 5th Floor, 1-23-3 Toranomon, Minato-ku, Tokyo,
105-6305, Japan

Unless specified otherwise, the conference information will be shared in Japan (JST timezone).

Sponsorship opportunities

Linux Plumbers Conference would not be possible without our sponsors. Many thanks to all the great organizations that have supported Linux Plumbers Conference over the years.

New sponsorship opportunities are available for 2025! We hope that your organization will consider joining our growing list of amazing sponsors this year. Find out more here.

LPC 2025 - Overview

Conference Details

The Linux Plumbers Conference is the premier event for developers working at all levels of the plumbing layer and beyond.

Taking place on Thursday 11th, Friday 12th and Saturday 13th of December, this year we will be both in person and remote (hybrid). However to minimize technical issues, we'd appreciate most of the content presenters being in-person. Taking place on Thursday 11th, Friday 12th and Saturday 13th of December, this year we will be both in person and remote (hybrid). However to minimize technical issues, we'd appreciate most of the content presenters being in-person.

The in-person venue is the Toranomon Hills Forum, Tokyo, Japan

Toranomon Hills Mori Tower 5th Floor, 1-23-3 Toranomon, Minato-ku, Tokyo, 105-6305, Japan

Unless specified otherwise, the conference information will be shared in Japan (JST timezone).

Conference Details

The Linux Plumbers Conference is the premier event for developers working at all levels of the plumbing layer and beyond.

東京 2025

LINUX PLUMBERS CONFERENCE

TOKYO, JAPAN / DECEMBER 11-13, 2025