Maintaining Ftrace

My unique way of doing things!

Introduction

- This is just how I do things

Introduction

- This is just how I do things
- I’m not suggesting you need to do any of this

Introduction

- This is just how I do things
- I’m not suggesting you need to do any of this
- I’m happy to hear about new tools

Introduction

- This is just how I do things
- I’m not suggesting you need to do any of this
- I’m happy to hear about new tools
- I’m very much grounded in this, so don’t expect me to do what you do!

Introduction

- This is just how I do things
- I’m not suggesting you need to do any of this
- I’m happy to hear about new tools
- I’m very much grounded in this, so don’t expect me to do what you do!
- This is all just for fun anyway 😉

My Server

Dell PowerEdge T430

- Got it from NewEgg
  - Refurbished
- 8 Bay 3.5" Swappable HDs
- 2x Intel Xeon E5-2683 v3 2.0GHz 14 Core Processors
- 256GB DDR4

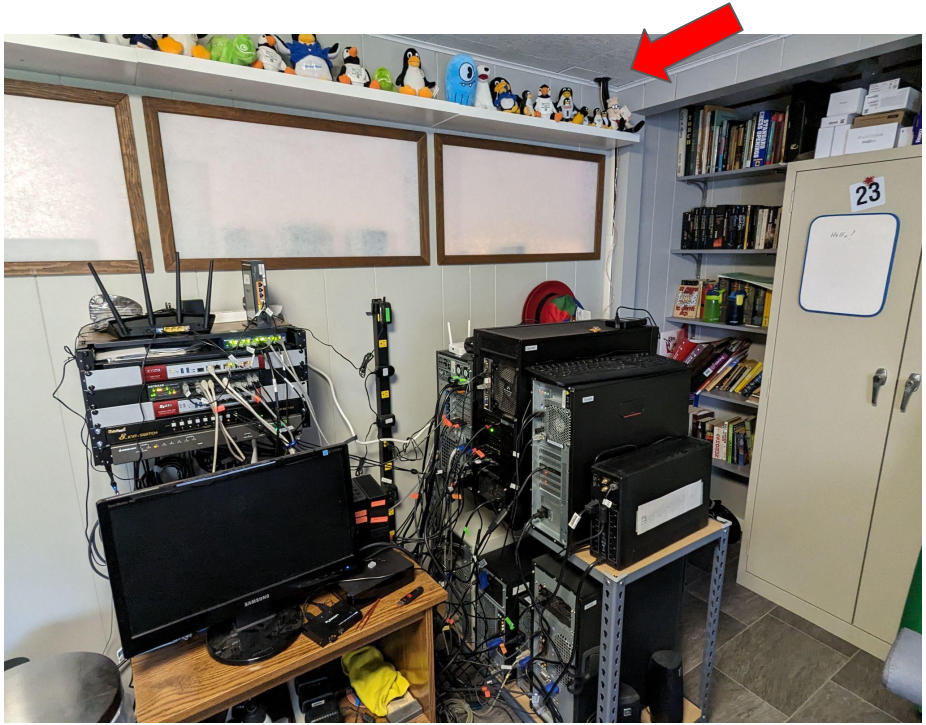
My Personal Workstation

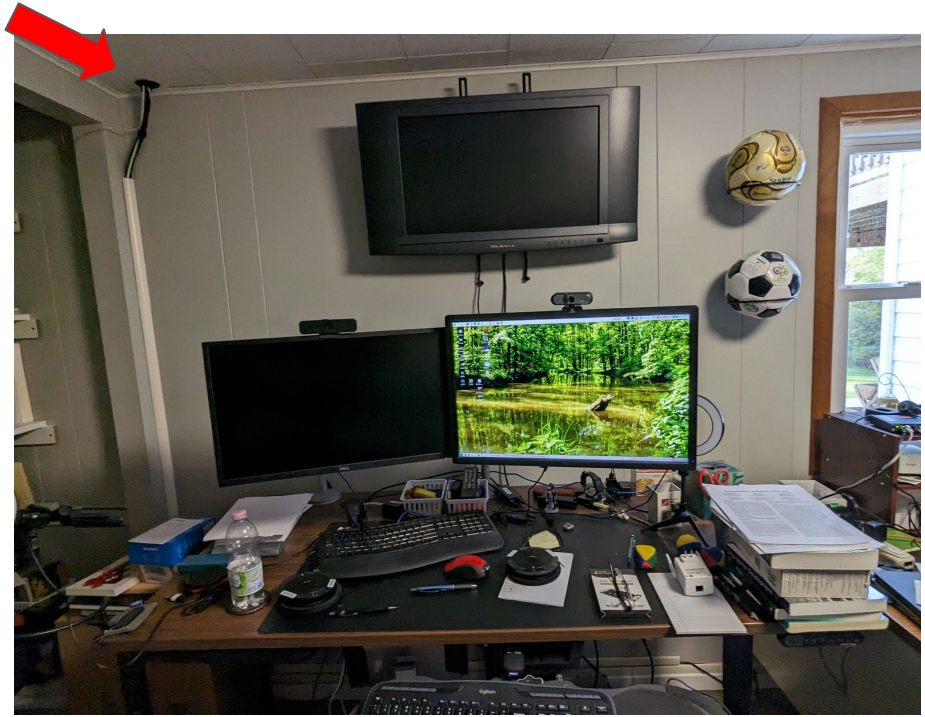
Put together myself

- Intel Xeon E5-2620 V4 Broadwell-EP 2.1 GHz
- 8 x 256KB L2 Cache
- 20MB L3 Cache
- 64GB (4 x 16GB) DDR4 SDRAM ECC

My Desk

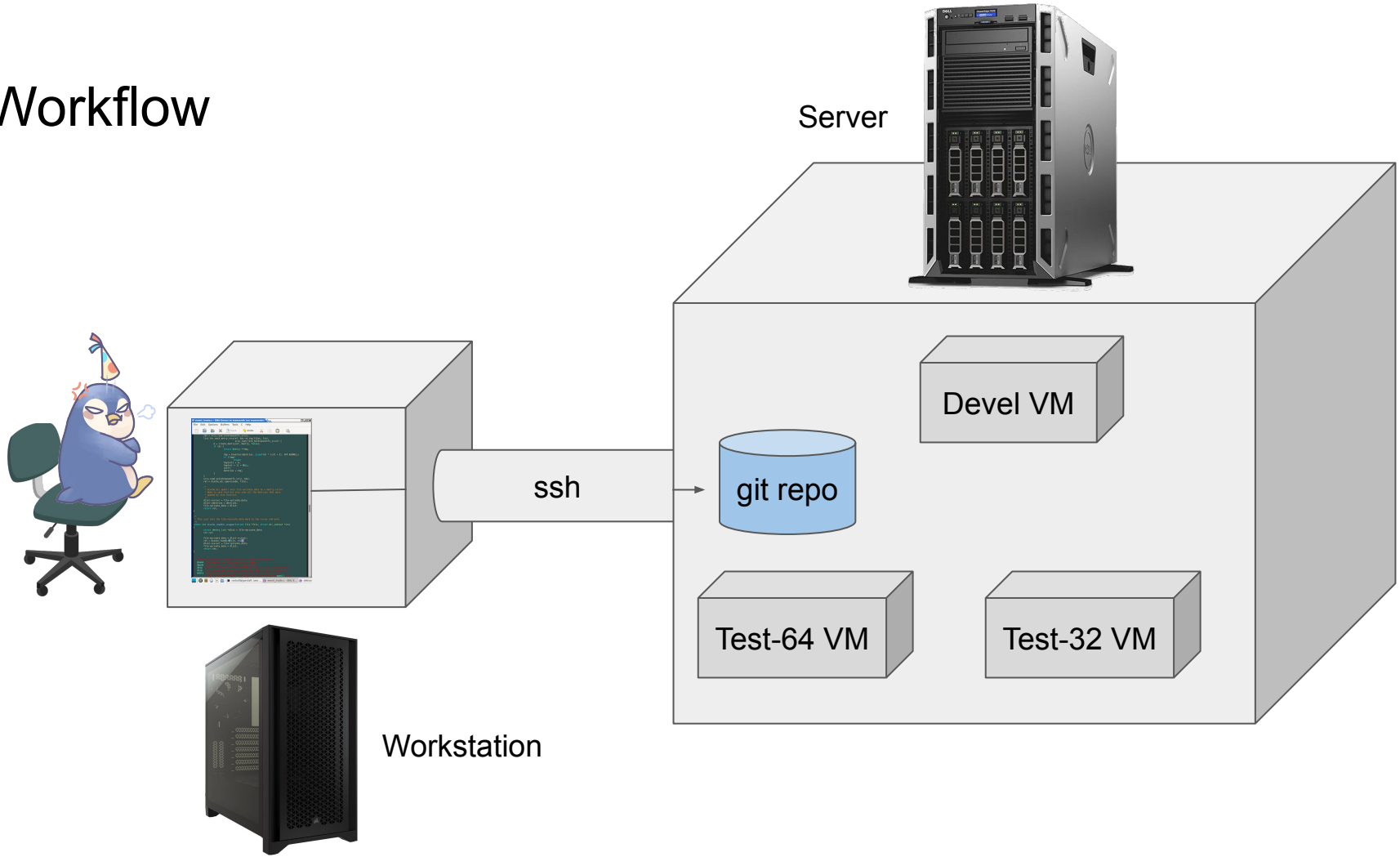
My Desk



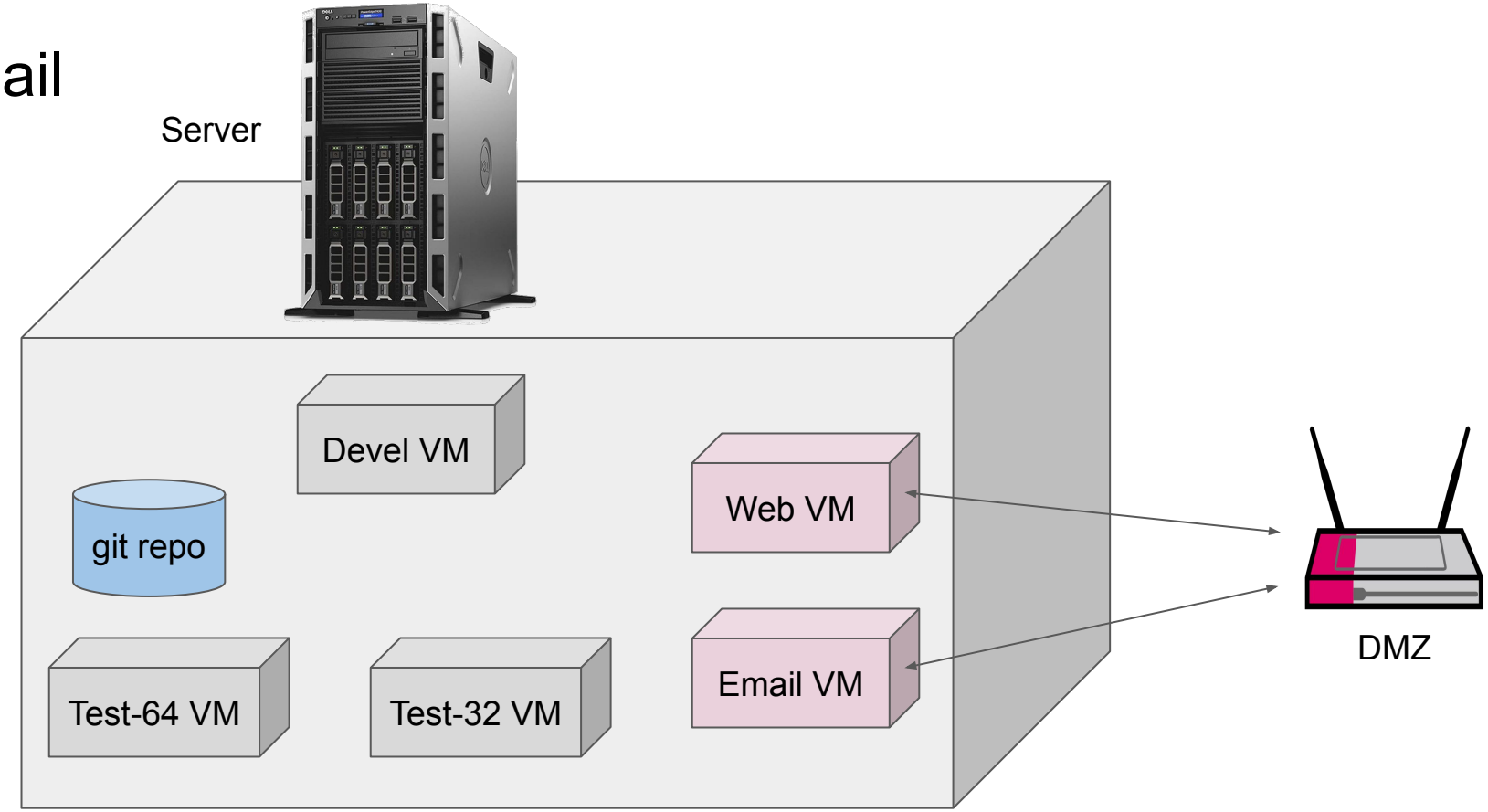


USB over CAT6

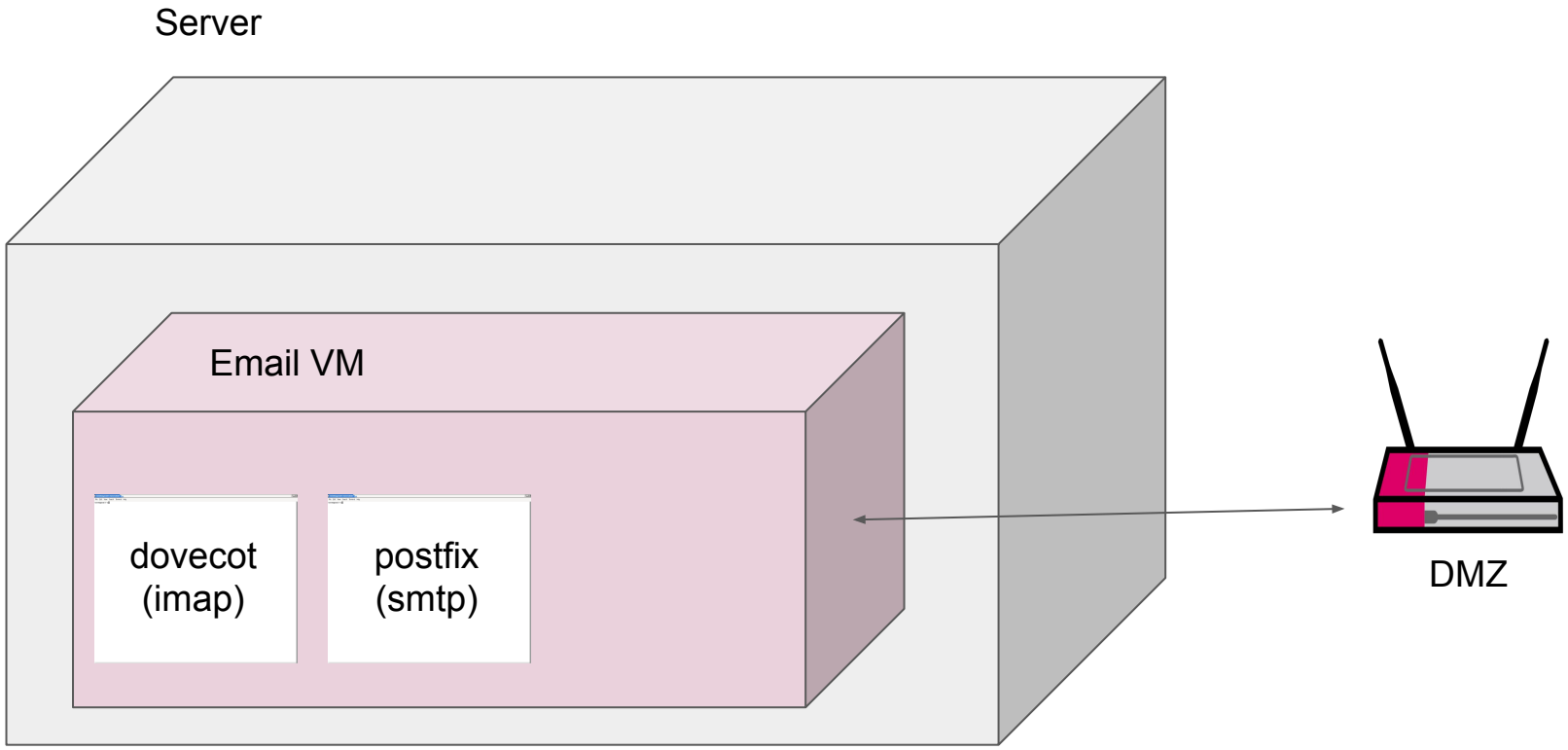
Workflow


Server
Devel VM
ssh
git repo
Test-64 VM
Test-32 VM
Workstation

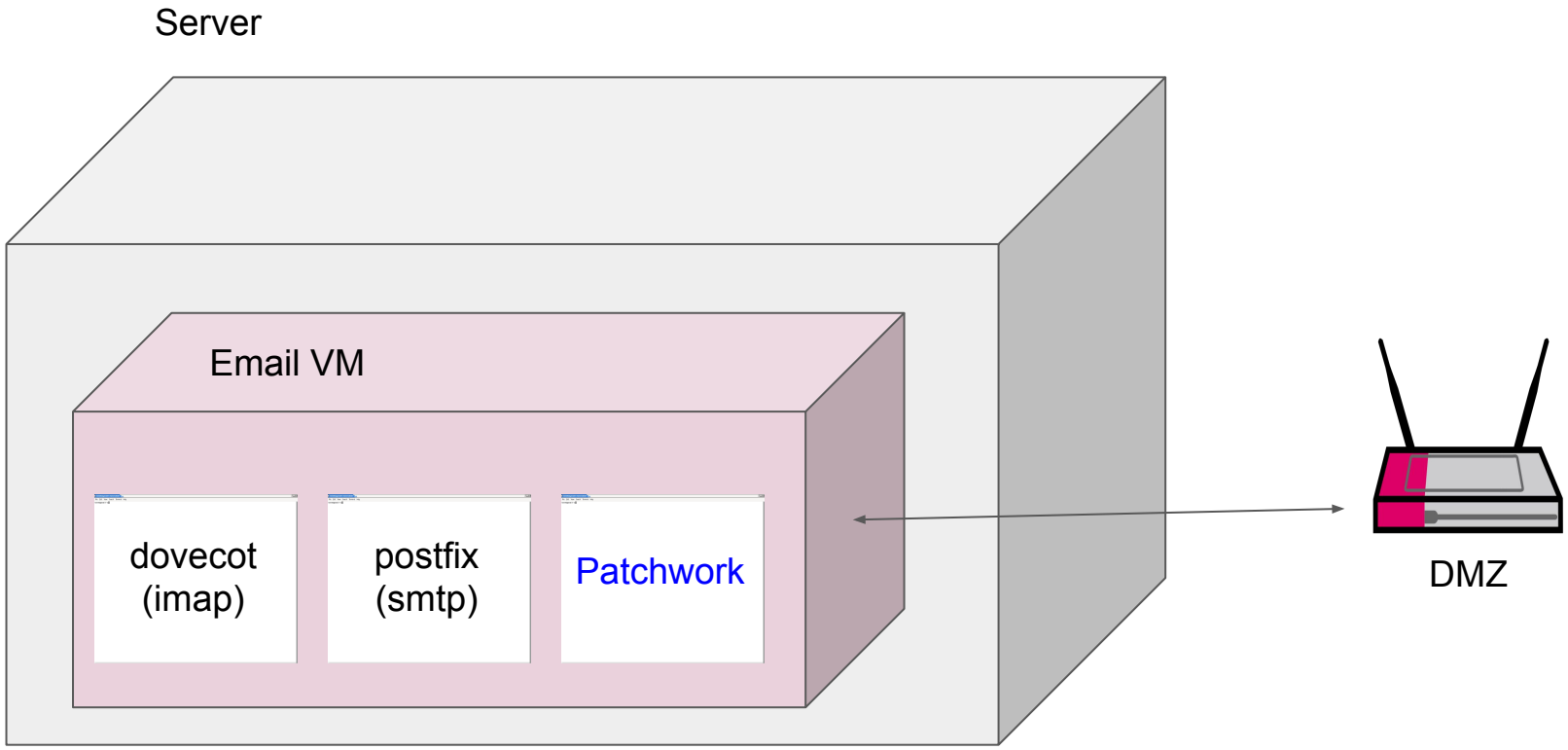
Email


Server
Devel VM
git repo
Web VM
Test-64 VM
Test-32 VM
Email VM
DMZ

Email



Email



Patchwork My INBOX Patches Bundles About this project 20 rostedt

Show patches with: State = **Action Required** | Archived = **No** | 2580 patches

« 1 2 3 4 … 25 26 »

Patch	Series	A/R/T	S/W/F	Date	Submitter	Delegate	State
sched: Filter root_task_group at the beginning	sched: Filter root_task_group at the beginning	- - -	- - -	2023-09-22	Haifeng Xu		New
[v2] sched: fix warning in bandwidth distribution	[v2] sched: fix warning in bandwidth distribution	- - -	- - -	2023-09-22	Josh Don		New
[printk,v1] printk: fix illegal pbufs access for !CONFIG_PRINTK	[printk,v1] printk: fix illegal pbufs access for !CONFIG_PRINTK	- 1 -	- - -	2023-09-20	John Ogness		New
[v2] sched/rt: move back to RT_GROUP_SCHED and rename it child	[v2] sched/rt: move back to RT_GROUP_SCHED and rename it child	- - -	- - -	2023-09-20	Yajun Deng		New
[2/2] sched: Update reference to sched_debug.c	sched: Small cleanups	- - -	- - -	2023-09-20	Sebastian Andrzej Siewior		New
[1/2] sched: Remove sysctl sched_child_runs_first.	sched: Small cleanups	- - -	- - -	2023-09-20	Sebastian Andrzej Siewior		New
printk/nbcon: Add assert that CPU migration is disabled when calling nbcon_context_try_acquire()	printk/nbcon: Add assert that CPU migration is disabled when calling nbcon_context_try_acquire()	- - -	- - -	2023-09-20	Petr Mladek		New
[printk,v2,11/11] lockdep: Add atomic write enforcement for lockdep splats	[printk,v2,01/11] printk: Make console_is_usable() available to nbcon	- - -	- - -	2023-09-19	John Ogness		New
[printk,v2,09/11] panic: Add atomic write enforcement to oops	[printk,v2,01/11] printk: Make console_is_usable() available to nbcon	- - -	- - -	2023-09-19	John Ogness		New
[printk,v2,08/11] panic: Add atomic write enforcement to warn/panic	[printk,v2,01/11] printk: Make console_is_usable() available to nbcon	- - -	- - -	2023-09-19	John Ogness		New
[printk,v2,07/11] printk: nbcon: Wire up nbcon into console_flush_all()	[printk,v2,01/11] printk: Make console_is_usable() available to nbcon	- - -	- - -	2023-09-19	John Ogness		New
[printk,v2,06/11] printk: nbcon: Wire up nbcon console atomic flushing	[printk,v2,01/11] printk: Make console_is_usable() available to nbcon	- - -	- - -	2023-09-19	John Ogness		New
[printk,v2,05/11] printk: nbcon: Provide function for atomic flushing	[printk,v2,01/11] printk: Make console_is_usable() available to nbcon	- - -	- - -	2023-09-19	John Ogness		New
[printk,v2,04/11] printk: nbcon: Provide functions to mark atomic write sections	[printk,v2,01/11] printk: Make console_is_usable() available to nbcon	- - -	- - -	2023-09-19	John Ogness		New
[printk,v2,03/11] printk: Add @flags argument for console_is_usable()	[printk,v2,01/11] printk: Make console_is_usable() available to nbcon	- 1 -	- - -	2023-09-19	John Ogness		New
[printk,v2,02/11] printk: Let console_is_usable() handle nbcon	[printk,v2,01/11] printk: Make console_is_usable() available to nbcon	- 1 -	- - -	2023-09-19	John Ogness		New
[printk,v2,01/11] printk: Make console_is_usable() available to nbcon	[printk,v2,01/11] printk: Make console_is_usable() available to nbcon	- 1 -	- - -	2023-09-19	John Ogness		New
[v3,3/3] doc: trusted-encrypted: add DCP as new trust source	DCP as trusted keys backend	- - -	- - -	2023-09-18	David Gstir		New
[v3,2/3] KEYS: trusted: Introduce support for NXP DCP-based trusted keys	DCP as trusted keys backend	- - -	- - -	2023-09-18	David Gstir		New
[v3,1/3] crypto: mxs-dcp: Add support for hardware provided keys	DCP as trusted keys backend	- - -	- - -	2023-09-18	David Gstir		New
[printk,v5,8/8] printk: nbcon: Allow drivers to mark unsafe regions and check state	provide nbcon base	- 1 -	- - -	2023-09-16	John Ogness		New
[printk,v5,7/8] printk: nbcon: Add emit function and callback function for atomic printing	provide nbcon base	- 1 -	- - -	2023-09-16	John Ogness		New
[printk,v5,6/8] printk: nbcon: Add sequence handling	provide nbcon base	- 1 -	- - -	2023-09-16	John Ogness		New
[printk,v5,5/8] printk: nbcon: Add ownership state functions	provide nbcon base	- 1 -	- - -	2023-09-16	John Ogness		New
[printk,v5,4/8] printk: nbcon: Add buffer management	provide nbcon base	- 1 -	- - -	2023-09-16	John Ogness		New
[printk,v5,3/8] printk: Make static printk buffers available to nbcon	provide nbcon base	- 1 -	- - -	2023-09-16	John Ogness		New
[printk,v5,2/8] printk: nbcon: Add acquire/release logic	provide nbcon base	- 1 -	- - -	2023-09-16	John Ogness		New
[printk,v5,1/8] printk: Add non-BKL (nbcon) console basic infrastructure	provide nbcon base	- 1 -	- - -	2023-09-16	John Ogness		New
selftests/user_events: Fix to unmount tracefs when test created mount	selftests/user_events: Fix to unmount tracefs when test created mount	- 1 -	- - -	2023-09-15	Beau Belgrave		New
[2/2] sched/eevdf: Use sched_attr::sched_runtime to set request/slice suggestion	sched/eevdf: sched_attr::sched_runtime slice hint	- - -	- - -	2023-09-15	Peter Zijlstra		New
[1/2] sched/eevdf: Also update slice on placement	sched/eevdf: sched_attr::sched_runtime slice hint	- - -	- - -	2023-09-15	Peter Zijlstra		New
[v2,2/2] static_call: Fix a wild-memory-access bug when static_call_key_sites(key) is true	static_call: Fix two wild-memory-access bugs in static_call_del_module()	- - -	- - -	2023-09-15	Jinjie Ruan		New

.procmailrc-patchwork

```
:0 Hc: mypatchwork.lock
* Subject: .*for-next
| /home/rostedt/bin/review-patch

:0 Hc: mypatchwork.lock
* Subject: .*for-linus
| /home/rostedt/bin/review-patch

:0 Hc
* Subject: .*PATCH
* !Subject: .*linus-commit
* !Subject: .*for-next
* !Subject: .*for-linus
{
        :0 H
        * !(To|CC): .*linux-kernel
        /dev/null

        :0 H
        * (To|CC): .*linux-trace-kernel
        /dev/null

        :0 EB
        * ^This is a note to let you know that I've just added the patch titled$
        /dev/null

        :0 c
        $MAILDIR/patchwork

        :0 : mypatchwork.lock
        | /home/rostedt/bin/supersede-patch
}
```

.procmailrc-patchwork

```
:0 Hc: mypatchwork.lock
* Subject: .*for-next
| /home/rostedt/bin/review-patch

:0 Hc: mypatchwork.lock
* Subject: .*for-linus
| /home/rostedt/bin/review-patch

:0 Hc
* Subject: .*PATCH
* !Subject: .*linus-commit
* !Subject: .*for-next
* !Subject: .*for-linus
{
        :0 H
        * !(To|CC): .*linux-kernel
        /dev/null

        :0 H
        * (To|CC): .*linux-trace-kernel
        /dev/null

        :0 EB
        * ^This is a note to let you know that I've just added the patch titled$
        /dev/null

        :0 c
        $MAILDIR/patchwork

        :0 : mypatchwork.lock
        | /home/rostedt/bin/supersede-patch
}
```

Reviewed

.procmailrc-patchwork

```
:0 Hc: mypatchwork.lock
* Subject: .*for-next
| /home/rostedt/bin/review-patch

:0 Hc: mypatchwork.lock
* Subject: .*for-linus
| /home/rostedt/bin/review-patch

:0 Hc
* Subject: .*PATCH
* !Subject: .*linus-commit
* !Subject: .*for-next
* !Subject: .*for-linus
{
        :0 H
        * !(To|CC): .*linux-kernel
        /dev/null

        :0 H
        * (To|CC): .*linux-trace-kernel
        /dev/null

        :0 EB
        * ^This is a note to let you know that I've just added the patch titled$
        /dev/null

        :0 c
        $MAILDIR/patchwork

        :0 : mypatchwork.lock
        | /home/rostedt/bin/supersede-patch
}
```

Subject Rules

.procmailrc-patchwork

```
:0 Hc: mypatchwork.lock
* Subject: .*for-next
| /home/rostedt/bin/review-patch

:0 Hc: mypatchwork.lock
* Subject: .*for-linus
| /home/rostedt/bin/review-patch

:0 Hc
* Subject: .*PATCH
* !Subject: .*linus-commit
* !Subject: .*for-next
* !Subject: .*for-linus
{
        :0 H
        * !(To|CC): .*linux-kernel
        /dev/null

        :0 H
        * (To|CC): .*linux-trace-kernel
        /dev/null

        :0 EB
        * ^This is a note to let you know that I've just added the patch titled$
        /dev/null

        :0 c
        $MAILDIR/patchwork

        :0 : mypatchwork.lock
        | /home/rostedt/bin/supersede-patch
}
```

Not sent to LKML?

.procmailrc-patchwork

```
:0 Hc: mypatchwork.lock
* Subject: .*for-next
| /home/rostedt/bin/review-patch

:0 Hc: mypatchwork.lock
* Subject: .*for-linus
| /home/rostedt/bin/review-patch

:0 Hc
* Subject: .*PATCH
* !Subject: .*linus-commit
* !Subject: .*for-next
* !Subject: .*for-linus
{
        :0 H
        * !(To|CC): .*linux-kernel
        /dev/null

        :0 H
        * (To|CC): .*linux-trace-kernel
        /dev/null

        :0 EB
        * ^This is a note to let you know that I've just added the patch titled$
        /dev/null

        :0 c
        $MAILDIR/patchwork

        :0 : mypatchwork.lock
        | /home/rostedt/bin/supersede-patch
}
```

Sent to linux-trace-kernel?

.procmailrc-patchwork

```
:0 Hc: mypatchwork.lock
* Subject: .*for-next
| /home/rostedt/bin/review-patch

:0 Hc: mypatchwork.lock
* Subject: .*for-linus
| /home/rostedt/bin/review-patch

:0 Hc
* Subject: .*PATCH
* !Subject: .*linus-commit
* !Subject: .*for-next
* !Subject: .*for-linus
{
        :0 H
        * !(To|CC): .*linux-kernel
        /dev/null

        :0 H
        * (To|CC): .*linux-trace-kernel
        /dev/null

        :0 EB
        * ^This is a note to let you know that I've just added the patch titled$
        /dev/null

        :0 c
        $MAILDIR/patchwork

        :0 : mypatchwork.lock
        | /home/rostedt/bin/supersede-patch
}
```

Ignore stable tree patches

.procmailrc-patchwork

```
:0 Hc: mypatchwork.lock
* Subject: .*for-next
| /home/rostedt/bin/review-patch

:0 Hc: mypatchwork.lock
* Subject: .*for-linus
| /home/rostedt/bin/review-patch

:0 Hc
* Subject: .*PATCH
* !Subject: .*linus-commit
* !Subject: .*for-next
* !Subject: .*for-linus
{
        :0 H
        * !(To|CC): .*linux-kernel
        /dev/null

        :0 H
        * (To|CC): .*linux-trace-kernel
        /dev/null

        :0 EB
        * ^This is a note to let you know that I've just added the patch titled$
        /dev/null

        :0 c
        $MAILDIR/patchwork

        :0 : mypatchwork.lock
        | /home/rostedt/bin/supersede-patch
}
```

Copy to “patchwork” folder

.procmailrc-patchwork

```
:0 Hc: mypatchwork.lock
* Subject: .*for-next
| /home/rostedt/bin/review-patch

:0 Hc: mypatchwork.lock
* Subject: .*for-linus
| /home/rostedt/bin/review-patch

:0 Hc
* Subject: .*PATCH
* !Subject: .*linus-commit
* !Subject: .*for-next
* !Subject: .*for-linus
{
        :0 H
        * !(To|CC): .*linux-kernel
        /dev/null

        :0 H
        * (To|CC): .*linux-trace-kernel
        /dev/null

        :0 EB
        * ^This is a note to let you know that I've just added the patch titled$
        /dev/null

        :0 c
        $MAILDIR/patchwork

        :0 : mypatchwork.lock
        | /home/rostedt/bin/supersede-patch
}
```

New version?

.procmailrc-linus

I subscribe to git-commits-head@vger.kernel.org!

.procmailrc-linus

I subscribe to git-commits-head@vger.kernel.org!

(Sends you an email for **every** commit Linus accepts!)

.procmailrc-linus

I subscribe to git-commits-head@vger.kernel.org!

(Sends you an email for **every** commit Linus accepts!)

```
:0 H
* List-id: <git-commits-head.vger.kernel.org>
{
        :0 c: mypatchwork.lock
        | /home/rostedt/bin/accept-patch-nofail

        :0 B
        * !Signed-off-by: Steven Rostedt
        /dev/null

        :0 fw
        | /bin/sed -e 's/^Subject: /Subject: [linus-commit]/'
}
```

.procmailrc-linus

I subscribe to git-commits-head@vger.kernel.org!

(Sends you an email for **every** commit Linus accepts!)

```
:0 H
* List-id: <git-commits-head.vger.kernel.org>
{
        :0 c: mypatchwork.lock
        | /home/rostedt/bin/accept-patch-nofail

        :0 B
        * !Signed-off-by: Steven Rostedt
        /dev/null

        :0 fw
        | /bin/sed -e 's/^Subject: /Subject: [linus-commit]/'
}
```

Accept all patches!

.procmailrc-linus

I subscribe to git-commits-head@vger.kernel.org!

(Sends you an email for **every** commit Linus accepts!)

```
:0 H
* List-id: <git-commits-head.vger.kernel.org>
{
        :0 c: mypatchwork.lock
        | /home/rostedt/bin/accept-patch-nofail

        :0 B
        * !Signed-off-by: Steven Rostedt
        /dev/null

        :0 fw
        | /bin/sed -e 's/^Subject: /Subject: [linus-commit]/'
}
```

Ignore?

.procmailrc-linus

I subscribe to git-commits-head@vger.kernel.org!

(Sends you an email for **every** commit Linus accepts!)

```
:0 H
* List-id: <git-commits-head.vger.kernel.org>
{
        :0 c: mypatchwork.lock
        | /home/rostedt/bin/accept-patch-nofail

        :0 B
        * !Signed-off-by: Steven Rostedt
        /dev/null

        :0 fw
        | /bin/sed -e 's/^Subject: /Subject: [linus-commit]/'
}
```

Add “[linus-commit]”

[linus-commit]x86/kprobes: Prohibit probing on compiler generated CFI checking code	Linux Kernel Mailing List	08/30/2023(Wed) 13:59	6.62KB
[linus-commit]tracing: Add back FORTIFY_SOURCE logic to kernel_stack event structure	Linux Kernel Mailing List	09/01/2023(Fri) 19:40	10.94KB
[linus-commit]ring_buffer: Use try_cmpxchg instead of cmpxchg	Linux Kernel Mailing List	09/01/2023(Fri) 19:40	6.16KB
[linus-commit]tracing: Remove unnecessary copying of tr->current_trace	Linux Kernel Mailing List	09/01/2023(Fri) 19:40	6.48KB
[linus-commit]tracing: Add free_trace_iter_content() helper function	Linux Kernel Mailing List	09/01/2023(Fri) 19:40	6.26KB
[linus-commit]tracing: Set actual size after ring buffer resize	Linux Kernel Mailing List	09/01/2023(Fri) 19:40	5.57KB
[linus-commit]tracing: Require all trace events to have a TRACE_SYSTEM	Linux Kernel Mailing List	09/01/2023(Fri) 19:40	5.49KB
[linus-commit]eventfs: Implement tracefs_inode_cache	Linux Kernel Mailing List	09/01/2023(Fri) 19:40	6.94KB
[linus-commit]tracefs: Rename and export some tracefs functions	Linux Kernel Mailing List	09/01/2023(Fri) 19:40	8.10KB
[linus-commit]eventfs: Implement eventfs dir creation functions	Linux Kernel Mailing List	09/01/2023(Fri) 19:40	12.97KB
[linus-commit]eventfs: Implement eventfs file add functions	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	8.48KB
[linus-commit]eventfs: Implement eventfs lookup, read, open functions	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	15.92KB
[linus-commit]eventfs: Implement functions to create files and dirs when accessed	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	10.24KB
[linus-commit]eventfs: Implement removal of meta data from eventfs	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	10.01KB
[linus-commit]eventfs: Move tracing/events to eventfs	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	16.80KB
[linus-commit]test: ftrace: Fix kprobe test for eventfs	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	6.76KB
[linus-commit]tracing/filters: Dynamically allocate filter_pred.regex	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	9.89KB
[linus-commit]tracing/filters: Enable filtering a cpumask field by another cpumask	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	10.21KB
[linus-commit]tracing/filters: Enable filtering a scalar field by a cpumask	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	9.94KB
[linus-commit]tracing/filters: Enable filtering the CPU common field by a cpumask	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	6.54KB
[linus-commit]tracing/filters: Optimise cpumask vs cpumask filtering when user mask is a single CPU	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	7.15KB
[linus-commit]tracing/filters: Optimise scalar vs cpumask filtering when the user mask is a single CPU	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	5.63KB
[linus-commit]tracing/filters: Optimise CPU vs cpumask filtering when the user mask is a single CPU	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	5.69KB
[linus-commit]tracing/filters: Further optimise scalar vs cpumask comparison	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	5.89KB
[linus-commit]tracing/filters: Document cpumask filtering	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	5.10KB
[linus-commit]tracing: Remove unused function declarations	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	5.21KB
[linus-commit]ftrace: Remove empty declaration ftrace_enable_daemon() and ftrace_disable_daemon()	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	5.02KB
[linus-commit]tracing/user_events: Optimize safe list traversals	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	6.61KB
[linus-commit]tracefs: Avoid changing i_mode to a temp value	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	5.62KB
[linus-commit]tracefs: Remove kerneldoc from struct eventfs_file	Linux Kernel Mailing List	09/01/2023(Fri) 19:41	5.54KB

Linux Kernel Tracing (linux-trace-kernel@vger.kernel.org)

Patchwork Linux Kernel Tracing Development | Patches | Bundles | About this project | Login | Register | Mail settings

Show patches with: State = **Action Required** | Archived = **No** | 119 patches

« 1 2 »

Patch	Series	A/R/T	S/W/F	Date	Submitter	Delegate	State
rtla: fix a example in rtla-timerlat-hist.rst	rtla: fix a example in rtla-timerlat-hist.rst	- - -	- - -	2023-09-19	Xie XiuQi		New
vmscan: add trace events for lru_gen	vmscan: add trace events for lru_gen	- - -	- - -	2023-09-19	Jaewon Kim		New
kprobes: Remove unnecessary initial values of variables	kprobes: Remove unnecessary initial values of variables	1 - -	- - -	2023-09-19	Li zeming		New
[v3,13/13] bpf: remove CONFIG_BPF_JIT dependency on CONFIG_MODULES of	mm: jit/text allocator	- - -	- - -	2023-09-18	Mike Rapoport		New
[v3,12/13] kprobes: remove dependency on CONFIG_MODULES	mm: jit/text allocator	- - -	- - -	2023-09-18	Mike Rapoport		New
[v3,11/13] x86/ftrace: enable dynamic ftrace without CONFIG_MODULES	mm: jit/text allocator	- - -	- - -	2023-09-18	Mike Rapoport		New
[v3,10/13] arch: make execmem setup available regardless of CONFIG_MODULES	mm: jit/text allocator	- - -	- - -	2023-09-18	Mike Rapoport		New
[v3,09/13] powerpc: extend execmem_params for kprobes allocations	mm: jit/text allocator	- - -	- - -	2023-09-18	Mike Rapoport		New
[v3,08/13] riscv: extend execmem_params for generated code allocations	mm: jit/text allocator	- - -	- - -	2023-09-18	Mike Rapoport		New
[v3,07/13] arm64, execmem: extend execmem_params for generated code allocations	mm: jit/text allocator	- - -	- - -	2023-09-18	Mike Rapoport		New
[v3,06/13] mm/execmem: introduce execmem_data_alloc()	mm: jit/text allocator	- - -	- - -	2023-09-18	Mike Rapoport		New
[v3,05/13] modules, execmem: drop module_alloc	mm: jit/text allocator	- - -	- - -	2023-09-18	Mike Rapoport		New
[v3,04/13] mm/execmem, arch: convert remaining overrides of module_alloc to execmem	mm: jit/text allocator	- - -	- - -	2023-09-18	Mike Rapoport		New
[v3,03/13] mm/execmem, arch: convert simple overrides of module_alloc to execmem	mm: jit/text allocator	- - -	- - -	2023-09-18	Mike Rapoport		New
[v3,02/13] mm: introduce execmem_text_alloc() and execmem_free()	mm: jit/text allocator	- - -	- - -	2023-09-18	Mike Rapoport		New
[v3,01/13] nios2: define virtual address space for modules	mm: jit/text allocator	2 - -	- - -	2023-09-18	Mike Rapoport		New
[3/9] mm/damon/core: use nr_accesses_bp as a source of damos_before_apply tracepoint	mm/damon: implement DAMOS apply intervals	- - -	- - -	2023-09-16	SeongJae Park		New
tracing/timerlat: Hotplug support for the user-space interface	tracing/timerlat: Hotplug support for the user-space interface	- - -	- - -	2023-09-15	Daniel Bristot de Oliveira		New
tools/rtla: Do not stop user-space if a cpu is offline	tools/rtla: Do not stop user-space if a cpu is offline	- - -	- - -	2023-09-15	Daniel Bristot de Oliveira		New

Patch	Series	A/R/T	S/W/F	Date	Submitter	Delegate	State
[v4,2/2] selftests/ftrace: Add new test case which checks non unique symbol	Return EADDRNOTAVAIL when func matches several symbols during kprobe creation	- - -	- - -	2023-08-25	Francis Laniel		New
[v4,1/2] tracing/kprobes: Return EADDRNOTAVAIL when func matches several symbols	Return EADDRNOTAVAIL when func matches several symbols during kprobe creation	- - -	- - -	2023-08-25	Francis Laniel		New
[v4] tracepoint: add new `tcp:tcp_ca_event` trace event	[v4] tracepoint: add new `tcp:tcp_ca_event` trace event	- - -	- - -	2023-08-25	Manjusaka		New
[RFC,v1,1/1] tracing/kprobes: Return ENAMESVRLSYMS when func matches several symbols	Return ENAMESVRLSYMS when func matches several symbols during PMU kprobe creation	- - -	- - -	2023-08-23	Francis Laniel		New
[v4,9/9] Documentation: tracing: Add a note about argument and retval access	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,8/9] Documentations: probes: Update fprobe document to use ftrace_regs	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,7/9] bpf: Enable kprobe_multi feature if CONFIG_FPROBE is enabled	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,6/9] tracing/fprobe: Enable fprobe events with CONFIG_DYNAMIC_FTRACE_WITH_ARGS	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	- - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,5/9] ftrace: Add ftrace_partial_regs() for converting ftrace_regs to pt_regs	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,4/9] fprobe: rethook: Use ftrace_regs in fprobe exit handler and rethook	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,3/9] tracing: Expose ftrace_regs regardless of CONFIG_FUNCTION_TRACER	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,2/9] fprobe: Use fprobe_regs in fprobe entry handler	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,1/9] Documentation: probes: Add a new ret_ip callback parameter	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v6,9/9] Documentation: tracing: Update fprobe event example with BTF field	tracing: Improbe BTF support on probe events	1 1 -	- - -	2023-08-22	Masami Hiramatsu (Google)		Queued
[v6,8/9] selftests/ftrace: Add BTF fields access testcases	tracing: Improbe BTF support on probe events	1 1 -	- - -	2023-08-22	Masami Hiramatsu (Google)		Queued
[v6,7/9] tracing/fprobe-event: Assume fprobe is a return event by $retval	tracing: Improbe BTF support on probe events	1 - -	- - -	2023-08-22	Masami Hiramatsu (Google)		Queued
[v6,6/9] tracing/probes: Add string type check with BTF	tracing: Improbe BTF support on probe events	1 - -	- - -	2023-08-22	Masami Hiramatsu (Google)		Queued

Patch	Series	A/R/T	S/W/F	Date	Submitter	Delegate	State
[v4,2/2] selftests/ftrace: Add new test case which checks non unique symbol	Return EADDRNOTAVAIL when func matches several symbols during kprobe creation	- - -	- - -	2023-08-25	Francis Laniel		New
[v4,1/2] tracing/kprobes: Return EADDRNOTAVAIL when func matches several symbols	Return EADDRNOTAVAIL when func matches several symbols during kprobe creation	- - -	- - -	2023-08-25	Francis Laniel		New
[v4] tracepoint: add new `tcp:tcp_ca_event` trace event	[v4] tracepoint: add new `tcp:tcp_ca_event` trace event	- - -	- - -	2023-08-25	Manjusaka		New
[RFC,v1,1/1] tracing/kprobes: Return ENAMESVRLSYMS when func matches several symbols	Return ENAMESVRLSYMS when func matches several symbols during PMU kprobe creation			23-08-23	Francis Laniel		New
[v4,9/9] Documentation: tracing: Add a note about argument and retval access	bpf: fprobe: rethook: Use ftrace_regs instead of pt_r		- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,8/9] Documentations: probes: Update fprobe document to use ftrace_regs	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,7/9] bpf: Enable kprobe_multi feature if CONFIG_FPROBE is enabled	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,6/9] tracing/fprobe: Enable fprobe events with CONFIG_DYNAMIC_FTRACE_WITH_ARGS	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	- - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,5/9] ftrace: Add ftrace_partial_regs() for converting ftrace_regs to pt_regs	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,4/9] fprobe: rethook: Use ftrace_regs in fprobe exit handler and rethook	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,3/9] tracing: Expose ftrace_regs regardless of CONFIG_FUNCTION_TRACER	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,2/9] fprobe: Use fprobe_regs in fprobe entry handler	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,1/9] Documentation: probes: Add a new ret_ip callback parameter	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v6,9/9] Documentation: tracing: Update fprobe event example with BTF field	tracing: Improbe BTF support on probe events	1 1 -	- - -	2023-08-22	Masami Hiramatsu (Google)		Queued
[v6,8/9] selftests/ftrace: Add BTF fields access testcases	tracing: Improbe BTF support on probe events	1 1 -	- - -	2023-08-22	Masami Hiramatsu (Google)		Queued
[v6,7/9] tracing/fprobe-event: Assume fprobe is a return event by $retval	tracing: Improbe BTF support on probe events	1 - -	- - -	2023-08-22	Masami Hiramatsu (Google)		Queued
[v6,6/9] tracing/probes: Add string type check with BTF	tracing: Improbe BTF support on probe events	1 - -	- - -	2023-08-22	Masami Hiramatsu (Google)		Queued

Click here

Patchwork Linux Kernel Tracing Development | Patches | Bundles | About this project | Login | Register | Mail settings

Show patches with: Series = **bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs** | State = **Action Required** | Archived = **No** | 9 patches

Patch	Series	A/R/T	S/W/F	Date	Submitter	Delegate	State
[v4,9/9] Documentation: tracing: Add a note about argument and retval access	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,8/9] Documentations: probes: Update fprobe document to use ftrace_regs	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,7/9] bpf: Enable kprobe_multi feature if CONFIG_FPROBE is enabled	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,6/9] tracing/fprobe: Enable fprobe events with CONFIG_DYNAMIC_FTRACE_WITH_ARGS	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	- - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,5/9] ftrace: Add ftrace_partial_regs() for converting ftrace_regs to pt_regs	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,4/9] fprobe: rethook: Use ftrace_regs in fprobe exit handler and rethook	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,3/9] tracing: Expose ftrace_regs regardless of CONFIG_FUNCTION_TRACER	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,2/9] fprobe: Use fprobe_regs in fprobe entry handler	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,1/9] Documentation: probes: Add a new ret_ip callback parameter	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New

Patchwork Linux Kernel Tracing Development | Patches | Bundles | About this project | Login | Register | Mail settings

Show patches with: Series = **bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs** | State = **Action Required** | Archived = **No** | 9 patches

Patch	Series	A/R/T	S/W/F	Date	Submitter	Delegate	State
[v4,9/9] Documentation: tracing: Add a note about argument and retval access	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,8/9] Documentations: probes: Update fprobe document to use ftrace_regs	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,7/9] bpf: Enable kprobe_multi feature if CONFIG_FPROBE is enabled	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,6/9] tracing/fprobe: Enable fprobe events with CONFIG_DYNAMIC_FTRACE_WITH_ARGS	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	- - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,5/9] ftrace: Add ftrace_partial_regs() for converting ftrace_regs to pt_regs	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,4/9] fprobe: rethook: Use ftrace_regs in fprobe exit handler and rethook	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,3/9] tracing: Expose ftrace_regs regardless of CONFIG_FUNCTION_TRACER	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,2/9] fprobe: Use fprobe_regs in fprobe entry handler	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New
[v4,1/9] Documentation: probes: Add a new ret_ip callback par	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs	1 - -	- - -	2023-08-23	Masami Hiramatsu (Google)		New

Click here

[v4,1/9] Documentation: probes: Add a new ret_ip callback parameter

13362751 diff mbox series

Message ID 169280373992.282662.14835192462715188987.stgit@devnote2 (mailing list archive)
State New
Headers show
Series bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs | expand

Commit Message

Masami Hiramatsu (Google) Aug. 23, 2023, 3:15 p.m. UTC

```
From: Masami Hiramatsu (Google) <mhiramat@kernel.org>

Add a new ret_ip callback parameter description.

Fixes: cb16330d1274 ("fprobe: Pass return address to the handlers")
Signed-off-by: Masami Hiramatsu (Google) <mhiramat@kernel.org>
---
 Changes in v4:
  - Update ret_ip description (Thanks Florent!)
---
 Documentation/trace/fprobe.rst |    8 ++++++--
 1 file changed, 6 insertions(+), 2 deletions(-)
```

Comments

Florent Revest Aug. 25, 2023, 4:11 p.m. UTC | #1

```
On Wed, Aug 23, 2023 at 5:15 PM Masami Hiramatsu (Google)
<mhiramat@kernel.org> wrote:
>
> From: Masami Hiramatsu (Google) <mhiramat@kernel.org>
>
> Add a new ret_ip callback parameter description.
>
> Fixes: cb16330d1274 ("fprobe: Pass return address to the handlers")
> Signed-off-by: Masami Hiramatsu (Google) <mhiramat@kernel.org>

Acked-by: Florent Revest <revest@chromium.org>
```

Patch

13362751 diff mbox series

```diff
diff --git a/Documentation/trace/fprobe.rst b/Documentation/trace/fprobe.rst
index 40dd2fbce861..5851a14eb893 100644
--- a/Documentation/trace/fprobe.rst
+++ b/Documentation/trace/fprobe.rst
@@ -91,9 +91,9 @@  The prototype of the entry/exit callback function are as follows:

 .. code-block:: c

- int entry_callback(struct fprobe *fp, unsigned long entry_ip, struct pt_regs *regs, void *entry_data);
+ int entry_callback(struct fprobe *fp, unsigned long entry_ip, unsigned long ret_ip, struct pt_regs *regs, void *entry_data);

- void exit_callback(struct fprobe *fp, unsigned long entry_ip, struct pt_regs *regs, void *entry_data);
+ void exit_callback(struct fprobe *fp, unsigned long entry_ip, unsigned long ret_ip, struct pt_regs *regs, void *entry_data);

 Note that the @entry_ip is saved at function entry and passed to exit handler.
 If the entry callback function returns !0, the corresponding exit callback will be cancelled.
@@ -108,6 +108,10 @@  If the entry callback function returns !0, the corresponding exit callback will
         Note that this may not be the actual entry address of the function but
         the address where the ftrace is instrumented.

+@ret_ip
+        This is the return address that the traced function will return to,
```

Patchwork Linux Kernel Tracing Development | Patches | Bundles | About this project | Register | Mail settings

Right Click here

[v4,1/9] Documentation: probes: Add a new ret_ip callback parameter

13362751 | diff | mbox | series

Message ID 169280373992.282662.14835192462715188987.stgit@devnote2 (mailing list archive)
State New
Headers show
Series bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs | expand

Commit Message

Masami Hiramatsu (Google) — Aug. 23, 2023, 3:15 p.m. UTC

```
From: Masami Hiramatsu (Google) <mhiramat@kernel.org>

Add a new ret_ip callback parameter description.

Fixes: cb16330d1274 ("fprobe: Pass return address to the handlers")
Signed-off-by: Masami Hiramatsu (Google) <mhiramat@kernel.org>
---
 Changes in v4:
  - Update ret_ip description (Thanks Florent!)
---
 Documentation/trace/fprobe.rst |    8 ++++++--
 1 file changed, 6 insertions(+), 2 deletions(-)
```

Comments

Florent Revest — Aug. 25, 2023, 4:11 p.m. UTC | #1

```
On Wed, Aug 23, 2023 at 5:15 PM Masami Hiramatsu (Google)
<mhiramat@kernel.org> wrote:
>
> From: Masami Hiramatsu (Google) <mhiramat@kernel.org>
>
> Add a new ret_ip callback parameter description.
>
> Fixes: cb16330d1274 ("fprobe: Pass return address to the handlers")
> Signed-off-by: Masami Hiramatsu (Google) <mhiramat@kernel.org>

Acked-by: Florent Revest <revest@chromium.org>
```

Patch

13362751 | diff | mbox | series

```diff
diff --git a/Documentation/trace/fprobe.rst b/Documentation/trace/fprobe.rst
index 40dd2fbce861..5851a14eb893 100644
--- a/Documentation/trace/fprobe.rst
+++ b/Documentation/trace/fprobe.rst
@@ -91,9 +91,9 @@  The prototype of the entry/exit callback function are as follows:

 .. code-block:: c

- int entry_callback(struct fprobe *fp, unsigned long entry_ip, struct pt_regs *regs, void *entry_data);
+ int entry_callback(struct fprobe *fp, unsigned long entry_ip, unsigned long ret_ip, struct pt_regs *regs, void *entry_data);

- void exit_callback(struct fprobe *fp, unsigned long entry_ip, struct pt_regs *regs, void *entry_data);
+ void exit_callback(struct fprobe *fp, unsigned long entry_ip, unsigned long ret_ip, struct pt_regs *regs, void *entry_data);

 Note that the @entry_ip is saved at function entry and passed to exit handler.
 If the entry callback function returns !0, the corresponding exit callback will be cancelled.
@@ -108,6 +108,10 @@  If the entry callback function returns !0, the corresponding exit callback will
         Note that this may not be the actual entry address of the function but
         the address where the ftrace is instrumented.

+@ret_ip
+        This is the return address that the traced function will return to,
```

Patchwork Linux Kernel Tracing Development | Patches | Bundles | About this project | Login | Register | Mail settings

[v4,1/9] Documentation: probes: Add a new ret_ip callback parameter

13362751 | diff | mbox | series

- Open Link in New Tab
- Open Link in New Window
- Open Link in New Private Window
- Bookmark Link...
- Save Link As...
- Save Link to Pocket
- Copy Link
- Search Google for "series"
- Inspect Accessibility Properties
- Inspect (Q)

Message ID	1692803739992.282662.14835192462715188987.stgit@devnote2 (mailing list archive)
State	New
Headers	show
Series	bpf: fprobe: rethook: Use ftrace_regs instead of pt_regs | expand

Commit Message

Masami Hiramatsu (Google)

```
From: Masami Hiramatsu (Google) <mhiramat@kernel.org>

Add a new ret_ip callback parameter description.

Fixes: cb16330d1274 ("fprobe: Pass return address to the handlers")
Signed-off-by: Masami Hiramatsu (Google) <mhiramat@kernel.org>
---
 Changes in v4:
  - Update ret_ip description (Thanks Florent!)
---
 Documentation/trace/fprobe.rst |    8 ++++++--
 1 file changed, 6 insertions(+), 2 deletions(-)
```

Comments

Florent Revest — Aug. 25, 2023, 4:11 p.m. UTC | #1

```
On Wed, Aug 23, 2023 at 5:15 PM Masami Hiramatsu (Google)
<mhiramat@kernel.org> wrote:
>
> From: Masami Hiramatsu (Google) <mhiramat@kernel.org>
>
> Add a new ret_ip callback parameter description.
>
> Fixes: cb16330d1274 ("fprobe: Pass return address to the handlers")
> Signed-off-by: Masami Hiramatsu (Google) <mhiramat@kernel.org>

Acked-by: Florent Revest <revest@chromium.org>
```

Patch

13362751 | diff | mbox | series

```diff
diff --git a/Documentation/trace/fprobe.rst b/Documentation/trace/fprobe.rst
index 40dd2fbce861..5851a14eb893 100644
--- a/Documentation/trace/fprobe.rst
+++ b/Documentation/trace/fprobe.rst
@@ -91,9 +91,9 @@  The prototype of the entry/exit callback function are as follows:

 .. code-block:: c

- int entry_callback(struct fprobe *fp, unsigned long entry_ip, struct pt_regs *regs, void *entry_data);
+ int entry_callback(struct fprobe *fp, unsigned long entry_ip, unsigned long ret_ip, struct pt_regs *regs, void *entry_data);

- void exit_callback(struct fprobe *fp, unsigned long entry_ip, struct pt_regs *regs, void *entry_data);
+ void exit_callback(struct fprobe *fp, unsigned long entry_ip, unsigned long ret_ip, struct pt_regs *regs, void *entry_data);

 Note that the @entry_ip is saved at function entry and passed to exit handler.
 If the entry callback function returns !0, the corresponding exit callback will be cancelled.
@@ -108,6 +108,10 @@  If the entry callback function returns !0, the corresponding exit callback will
         Note that this may not be the actual entry address of the function but
         the address where the ftrace is instrumented.

+@ret_ip: function will return to,
```

https://patchwork.kernel.org/series/778651/mbox/

Download the patch series

```
>$ wget -O /tmp/ftrace.patch https://patchwork.kernel.org/series/778651/mbox/
```

Download the patch series

```
>$ wget -O /tmp/ftrace.patch https://patchwork.kernel.org/series/778651/mbox/

>$ add-links.pl /tmp/ftrace.{patch,mbox}
Link: https://lkml.kernel.org/r/169280373992.282662.14835192462715188987.stgit@devnote2
Link: https://lkml.kernel.org/r/169280375109.282662.4109179404470188137.stgit@devnote2
Link: https://lkml.kernel.org/r/169280376296.282662.3179053699894615088.stgit@devnote2
Link: https://lkml.kernel.org/r/169280377434.282662.7610009313268953247.stgit@devnote2
Link: https://lkml.kernel.org/r/169280378611.282662.4078983611827223131.stgit@devnote2
Link: https://lkml.kernel.org/r/169280379741.282662.12221517584561036597.stgit@devnote2
Link: https://lkml.kernel.org/r/169280380720.282662.17571417296398071399.stgit@devnote2
Link: https://lkml.kernel.org/r/169280381726.282662.9429943163047257398.stgit@devnote2
Link: https://lkml.kernel.org/r/169280382895.282662.14910495061790007288.stgit@devnote2
```

Download the patch series

```
>$ wget -O /tmp/ftrace.patch https://patchwork.kernel.org/series/778651/mbox/

>$ add-links.pl /tmp/ftrace.{patch,mbox}
Link: https://lkml.kernel.org/r/169280373992.282662.14835192462715188987.stgit@devnote2
Link: https://lkml.kernel.org/r/169280375109.282662.4109179404470188137.stgit@devnote2
Link: https://lkml.kernel.org/r/169280376296.282662.3179053699894615088.stgit@devnote2
Link: https://lkml.kernel.org/r/169280377434.282662.7610009313268953247.stgit@devnote2
Link: https://lkml.kernel.org/r/169280378611.282662.4078983611827223131.stgit@devnote2
Link: https://lkml.kernel.org/r/169280379741.282662.12221517584561036597.stgit@devnote2
Link: https://lkml.kernel.org/r/169280380720.282662.17571417296398071399.stgit@devnote2
Link: https://lkml.kernel.org/r/169280381726.282662.9429943163047257398.stgit@devnote2
Link: https://lkml.kernel.org/r/169280382895.282662.14910495061790007288.stgit@devnote2

>$ git am -s /tmp/ftrace.mbox
Link: https://lkml.kernel.org/r/169280375109.282662.4109179404470188137.stgit@devnote2
Applying: Documentation: probes: Add a new ret_ip callback parameter
Applying: fprobe: Use fprobe_regs in fprobe entry handler
Applying: tracing: Expose ftrace_regs regardless of CONFIG_FUNCTION_TRACER
Applying: fprobe: rethook: Use ftrace_regs in fprobe exit handler and rethook
Applying: ftrace: Add ftrace_partial_regs() for converting ftrace_regs to pt_regs
Applying: tracing/fprobe: Enable fprobe events with CONFIG_DYNAMIC_FTRACE_WITH_ARGS
Applying: bpf: Enable kprobe_multi feature if CONFIG_FPROBE is enabled
Applying: Documentations: probes: Update fprobe document to use ftrace_regs
error: patch failed: Documentation/trace/fprobe.rst:112
error: Documentation/trace/fprobe.rst: patch does not apply
Patch failed at 0008 Documentations: probes: Update fprobe document to use ftrace_regs
hint: Use 'git am --show-current-patch=diff' to see the failed patch
When you have resolved this problem, run "git am --continue".
If you prefer to skip this patch, run "git am --skip" instead.
To restore the original branch and stop patching, run "git am --abort".
```

Using quilt to fix git am errors

```
>$ git am --show-current-patch=diff > /tmp/ftrace.patch
```

Using quilt to fix git am errors

```
>$ git am --show-current-patch=diff > /tmp/ftrace.patch

>$ quilt del ftrace.patch
```

Using quilt to fix git am errors

```
>$ git am --show-current-patch=diff > /tmp/ftrace.patch

>$ quilt del ftrace.patch

>$ quilt import /tmp/ftrace.patch
```

Using quilt to fix git am errors

```
>$ git am --show-current-patch=diff > /tmp/ftrace.patch

>$ quilt del ftrace.patch

>$ quilt import /tmp/ftrace.patch

>$ quilt push -f
Applying patch ftrace.patch
patching file Documentation/trace/fprobe.rst
Hunk #2 FAILED at 112.
1 out of 2 hunks FAILED -- saving rejects to file Documentation/trace/fprobe.rst.rej
Applied patch ftrace.patch (forced; needs refresh)
```

Using quilt to fix git am errors

```
>$ git am --show-current-patch=diff > /tmp/ftrace.patch

>$ quilt del ftrace.patch

>$ quilt import /tmp/ftrace.patch

>$ quilt push -f
Applying patch ftrace.patch
patching file Documentation/trace
Hunk #2 FAILED at 112.
1 out of 2 hunks FAILED -- saving rejects to  e Documentation/trace/fprobe.rst.rej
Applied patch ftrace.patch (forced; needs refresh)
```

Remember files

File Edit Options Buffers Tools Text Help

Save Undo

```
-- Documentation/trace/fprobe.rst
+++ Documentation/trace/fprobe.rst
@@ -112,12 +112,10 @@ If the entry callback function returns !0, the corresponding exit callback will
         This is the return address that the traced function will return to,
         somewhere in the caller. This can be used at both entry and exit.

-@regs
-        This is the `pt_regs` data structure at the entry and exit. Note that
-        the instruction pointer of @regs may be different from the @entry_ip
-        in the entry_handler. If you need traced instruction pointer, you need
-        to use @entry_ip. On the other hand, in the exit_handler, the instruction
-        pointer of @regs is set to the currect return address.
+@fregs
+        This is the `ftrace_regs` data structure at the entry and exit. Note that
+        the instruction pointer of @fregs may be incorrect in entry handler and
+        exit handler, so you have to use @entry_ip and @ret_ip instead.

 @entry_data
         This is a local storage to share the data between entry and exit handlers.
```

-:--@ **fprobe.rst.rej</ssh:rostedt@rostedt.org:>** All L1 (Diff)

```
        Note that this may not be the actual entry address of the function but
        the address where the ftrace is instrumented.

@ret_ip
        This is the return address that the traced function will return to,
        somewhere in the caller. This can be used at both entry and exit.

@regs
        This is the `pt_regs` data structure at the entry and exit. Note that
        the instruction pointer of @regs may be different from the @entry_ip
        in the entry_handler. If you need traced instruction pointer, you need
        to use @entry_ip. On the other hand, in the exit_handler, the instruction
        pointer of @regs is set to the current return address.

@entry_data
        This is a local storage to share the data between entry and exit handlers.
        This storage is NULL by default. If the user specify `exit_handler` field
        and `entry_data_size` field when registering the fprobe, the storage is
        allocated and passed to both `entry_handler` and `exit_handler`.

Share the callbacks with kprobes
```

-:--@ **fprobe.rst</ssh:rostedt@rostedt.org:>** 52% L109 (ReST)

File Edit Options Buffers Tools Help

Save Undo

```
        This is the return address that the traced function will return to,
        somewhere in the caller. This can be used at both entry and exit.

-@regs
-       This is the `pt_regs` data structure at the entry and exit. Note that
-       the instruction pointer of @regs may be different from the @entry_ip
-       in the entry_handler. If you need traced instruction pointer, you need
-       to use @entry_ip. On the other hand, in the exit_handler, the instruction
-       pointer of @regs is set to the currect return address.
+@fregs
+       This is the `ftrace_regs` data structure at the entry and exit. Note that
+       the instruction pointer of @fregs may be incorrect in entry handler and
+       exit handler, so you have to use @entry_ip and @ret_ip instead.

 @entry_data
        This is a local storage to share the data between entry and exit handlers.
```

1:**@ **a** All L14 (Fundamental)

File Edit Options Buffers Tools Help

Save Undo

```
        This is the return address that the traced function will return to,
        somewhere in the caller. This can be used at both entry and exit.

-@regs
-       This is the `pt_regs` data structure at the entry and exit. Note that
-       the instruction pointer of @regs may be different from the @entry_ip
-       in the entry_handler. If you need traced instruction pointer, you need
-       to use @entry_ip. On the other hand, in the exit_handler, the instruction
-       pointer of @regs is set to the currect return address.

@entry_data
        This is a local storage to share the data between entry and exit handlers.
```

1:**@ a All L12 (Fundamental)

File Edit Options Buffers Tools Help

Save Undo

```
        This is the return address that the traced function will return to,
        somewhere in the caller. This can be used at both entry and exit.

-@regs
-       This is the `pt_regs` data structure at the entry and exit. Note that
-       the instruction pointer of @regs may be different from the @entry_ip
-       in the entry_handl  If you need traced instruction pointer, you need
-       to use @ent  he other hand, in the exit_handler, the instruction
-       poin  to the currect return address.

@ent
        s is a local storage to share the data between entry and exit handlers.
```

Ctrl^X r k

1:**@ a All L12 (Fundamental)

File Edit Options Buffers Tools Help

Save Undo

```
        This is the return address that the traced function will return to,
        somewhere in the caller. This can be used at both entry and exit.

@regs
        This is the `pt_regs` data structure at the entry and exit. Note that
        the instruction pointer of @regs may be different from the @entry_ip
        in the entry_handler. If you need traced instruction pointer, you need
        to use @entry_ip. On the other hand, in the exit_handler, the instruction
        pointer of @regs is set to the currect return address.

@entry_data
        This is a local storage to share the data between entry and exit handlers.
```

1:**@ **a** All L12 (Fundamental)

File Edit Options Buffers Tools Help

Save Undo

```
        This is the return address that the traced function will return to,
        somewhere in the caller. This can be used at both entry and exit.

@regs
        This is the `pt_regs` data structure at the entry and exit. Note that
        the instruction pointer of @regs may be different from the @entry_ip
        in the entry_handler. If you need traced instruction pointer, you need
        to use @entry_ip. On the other hand, in the exit_handler, the instruction
        pointer of @regs is set to the currect return address.

@entry_data
        This is a local storage to share the data between entry and exit handlers.
```

1:**@ **a** All L12 (Fundamental)

```
        This is the return address that the traced function will return to,
        somewhere in the caller. This can be used at both entry and exit.

@regs
        This is the `pt_regs` data structure at the entry and exit. Note that
        the instruction pointer of @regs may be different from the @entry_ip
        in the entry_handler. If you need traced instruction pointer, you need
        to use @entry_ip. On the other hand, in the exit_handler, the instruction
        pointer of @regs is set to the current return addresso.

@entry_data
        This is a local storage to share the data between entry and exit handlers.
```

1:**@ **b** All L13 (Fundamental)

End of buffer

File Edit Options Buffers Tools Minibuf Help

Save Undo

```
        This is the return address that the traced function will return to,
        somewhere in the caller. This can be used at both entry and exit.

@regs
        This is the `pt_regs` data structure at the entry and exit. Note that
        the instruction pointer of @regs may be different from the @entry_ip
        in the entry_handler. If you need traced instruction pointer, you need
        to use @entry_ip. On the other hand, in the exit_handler, the instruction
        pointer of @regs is set to the current return addresso.

@entry_data
        This is a local storage to share the data between entry and exit handlers.
```

A: 1:**@ b All L9 (Fundamental)

```
        This is the return address that the traced function will return to,
        somewhere in the caller. This can be used at both entry and exit.

@regs
        This is the `pt_regs` data structure at the entry and exit. Note that
        the instruction pointer of @regs may be different from the @entry_ip
        in the entry_handler. If you need traced instruction pointer, you need
        to use @entry_ip. On the other hand, in the exit_handler, the instruction
        pointer of @regs is set to the currect return address.

@entry_data
        This is a local storage to share the data between entry and exit handlers.
```

B: 1:**@ a All L12 (Fundamental)

M-x ediff-buffers

File Edit Options Buffers Tools Text Help

Save Undo

```
--- Documentation/trace/fprobe.rst
+++ Documentation/trace/fprobe.rst
@@ -112,12 +112,10 @@ If the entry callback function returns !0, the corresponding exit callback will
         This is the return address that the traced function will return to,
         somewhere in the caller. This can be used at both entry and exit.

-@regs
-        This is the `pt_regs` data structure at the entry and exit. Note that
-        the instruction pointer of @regs may be different from the @entry_ip
-        in the entry_handler. If you need traced instruction pointer, you need
-        to use @entry_ip. On the other hand, in the exit_handler, the instruction
-        pointer of @regs is set to the currect return address.
+@fregs
+        This is the `ftrace_regs` data structure at the entry and exit. Note that
+        the instruction pointer of @fregs may be incorrect in entry handler and
+        exit handler, so you have to use @entry_ip and @ret_ip instead.

 @entry_data
         This is a local storage to share the data between entry and exit handlers.
```

-:--@ fprobe.rst.rej</ssh:rostedt@rostedt.org:> All L17 (Diff)

```
        Note that this may not be the actual entry address of the function but
        the address where the ftrace is instrumented.

@ret_ip
        This is the return address that the traced function will return to,
        somewhere in the caller. This can be used at both entry and exit.

+@fregs
+        This is the `ftrace_regs` data structure at the entry and exit. Note that
+        the instruction pointer of @fregs may be incorrect in entry handler and
+        exit handler, so you have to use @entry_ip and @ret_ip instead.
@regs
        This is the `pt_regs` data structure at the entry and exit. Note that
        the instruction pointer of @regs may be different from the @entry_ip
        in the entry_handler. If you need traced instruction pointer, you need
        to use @entry_ip. On the other hand, in the exit_handler, the instruction
        pointer of @regs is set to the current return address.

@entry_data
        This is a local storage to share the data between entry and exit handlers.
        This storage is NULL by default. If the user specify `exit_handler` field
```

-:**@ fprobe.rst</ssh:rostedt@rostedt.org:> 50% L118 (ReST)

Auto-saving...done

File Edit Options Buffers Tools Text Help

Save Undo

```
--- Documentation/trace/fprobe.rst
+++ Documentation/trace/fprobe.rst
@@ -112,12 +112,10 @@ If the entry callback function returns !0, the corresponding exit callback will
         This is the return address that the traced function will return to,
         somewhere in the caller. This can be used at both entry and exit.

-@regs
-        This is the `pt_regs` data structure at the entry and exit. Note that
-        the instruction pointer of @regs may be different from the @entry_ip
-        in the entry_handler. If you need traced instruction pointer, you need
-        to use @entry_ip. On the other hand, in the exit_handler, the instruction
-        pointer of @regs is set to the currect return address.
+@fregs
+        This is the `ftrace_regs` data structure at the entry and exit. Note that
+        the instruction pointer of @fregs may be incorrect in entry handler and
+        exit handler, so you have to use @entry_ip and @ret_ip instead.

 @entry_data
         This is a local storage to share the data between entry and exit handlers.
```

-:--@ **fprobe.rst.rej</ssh:rostedt@rostedt.org:>** All L17 (Diff)

```
        Note that this may not be the actual entry address of the function but
        the address where the ftrace is instrumented.

@ret_ip
        This is the return address that the traced function will return to,
        somewhere in the caller. This can be used at both entry and exit.

+@fregs
+        This is the `ftrace_regs` data structure at the entry and exit. Note that
+        the instruction pointer of @fregs may be incorrect in entry handler and
+        exit handler, so you have to use @entry_ip and @ret_ip instead.
@regs
        This is the `pt_regs` data structure at the entry and exit. Note that
        the instruction pointer of @regs may be different from the @entry_ip
        in the entry_handler. If you need traced instruction pointer, you need
        to use @entry_ip. On the other hand, in the exit_handler, the instruction
        pointer of @regs is set to the current return address.

@entry_data
        This is a local storage to share the data between entry and exit handlers.
        This storage is NULL by default. If the user specify `exit_handler` field
```

-:**@ **fprobe.rst</ssh:rostedt@rostedt.org:>** 50% L118 (ReST)

Auto-saving...done

Ctrl^X r k

File Edit Options Buffers Tools Text Help

Save Undo

```
--- Documentation/trace/fprobe.rst
+++ Documentation/trace/fprobe.rst
@@ -112,12 +112,10 @@ If the entry callback function returns !0, the corresponding exit callback will
         This is the return address that the traced function will return to,
         somewhere in the caller. This can be used at both entry and exit.

-@regs
-        This is the `pt_regs` data structure at the entry and exit. Note that
-        the instruction pointer of @regs may be different from the @entry_ip
-        in the entry_handler. If you need traced instruction pointer, you need
-        to use @entry_ip. On the other hand, in the exit_handler, the instruction
-        pointer of @regs is set to the currect return address.
+@fregs
+        This is the `ftrace_regs` data structure at the entry and exit. Note that
+        the instruction pointer of @fregs may be incorrect in entry handler and
+        exit handler, so you have to use @entry_ip and @ret_ip instead.

 @entry_data
         This is a local storage to share the data between entry and exit handlers.
```

-:--@ **fprobe.rst.rej</ssh:rostedt@rostedt.org:>** All L17 (Diff)

```
        Note that this may not be the actual entry address of the function but
        the address where the ftrace is instrumented.

@ret_ip
        This is the return address that the traced function will return to,
        somewhere in the caller. This can be used at both entry and exit.

@fregs
        This is the `ftrace_regs` data structure at the entry and exit. Note that
        the instruction pointer of @fregs may be incorrect in entry handler and
        exit handler, so you have to use @entry_ip and @ret_ip instead.
@regs
        This is the `pt_regs` data structure at the entry and exit. Note that
        the instruction pointer of @regs may be different from the @entry_ip
        in the entry_handler. If you need traced instruction pointer, you need
        to use @entry_ip. On the other hand, in the exit_handler, the instruction
        pointer of @regs is set to the current return address.

@entry_data
        This is a local storage to share the data between entry and exit handlers.
        This storage is NULL by default. If the user specify `exit_handler` field
```

-:**@ **fprobe.rst</ssh:rostedt@rostedt.org:>** 50% L125 (ReST)

File Edit Options Buffers Tools Text Help

Save Undo

```
--- Documentation/trace/fprobe.rst
+++ Documentation/trace/fprobe.rst
@@ -112,12 +112,10 @@ If the entry callback function returns !0, the corresponding exit callback will
         This is the return address that the traced function will return to,
         somewhere in the caller. This can be used at both entry and exit.

-@regs
-        This is the `pt_regs` data structure at the entry and exit. Note that
-        the instruction pointer of @regs may be different from the @entry_ip
-        in the entry_handler. If you need traced instruction pointer, you need
-        to use @entry_ip. On the other hand, in the exit_handler, the instruction
-        pointer of @regs is set to the currect return address.
+@fregs
+        This is the `ftrace_regs` data structure at the entry and exit. Note that
+        the instruction pointer of @fregs may be incorrect in entry handler and
+        exit handler, so you have to use @entry_ip and @ret_ip instead.

 @entry_data
         This is a local storage to share the data between entry and exit handlers.
```

-:--@ fprobe.rst.rej</ssh:rostedt@rostedt.org:> All L17 (Diff)

```
        Note that this may not be the actual entry address of the function but
        the address where the ftrace is instrumented.

@ret_ip
        This is the return address that the traced function will return to,
        somewhere in the caller. This can be used at both entry and exit.

@fregs
        This is the `ftrace_regs` data structure at the entry and exit. Note that
        the instruction pointer of @fregs may be incorrect in entry handler and
        exit handler, so you have to use @entry_ip and @ret_ip instead.

@entry_data
        This is a local storage to share the data between entry and exit handlers.
        This storage is NULL by default. If the user specify `exit_handler` field
        and `entry_data_size` field when registering the fprobe, the storage is
        allocated and passed to both `entry_handler` and `exit_handler`.

Share the callbacks with kprobes
================================
```

-:**@ fprobe.rst</ssh:rostedt@rostedt.org:> 53% L119 (ReST)

Check the changes

```
>$ git diff
diff --git a/Documentation/trace/fprobe.rst b/Documentation/trace/fprobe.rst
index 196f52386aaa..64ef522f7a64 100644
--- a/Documentation/trace/fprobe.rst
+++ b/Documentation/trace/fprobe.rst
@@ -91,9 +91,9 @@ The prototype of the entry/exit callback function are as follows:

 .. code-block:: c

- int entry_callback(struct fprobe *fp, unsigned long entry_ip, unsigned long ret_ip, struct pt_regs *regs, void *entry_data);
+ int entry_callback(struct fprobe *fp, unsigned long entry_ip, unsigned long ret_ip, struct ftrace_regs *fregs, void *entry_data);

- void exit_callback(struct fprobe *fp, unsigned long entry_ip, unsigned long ret_ip, struct pt_regs *regs, void *entry_data);
+ void exit_callback(struct fprobe *fp, unsigned long entry_ip, unsigned long ret_ip, struct ftrace_regs *fregs, void *entry_data);

 Note that the @entry_ip is saved at function entry and passed to exit handler.
 If the entry callback function returns !0, the corresponding exit callback will be cancelled.
@@ -112,12 +112,10 @@ If the entry callback function returns !0, the corresponding exit callback will
         This is the return address that the traced function will return to,
         somewhere in the caller. This can be used at both entry and exit.

-@regs
-        This is the `pt_regs` data structure at the entry and exit. Note that
-        the instruction pointer of @regs may be different from the @entry_ip
-        in the entry_handler. If you need traced instruction pointer, you need
-        to use @entry_ip. On the other hand, in the exit_handler, the instruction
-        pointer of @regs is set to the current return address.
+@fregs
+        This is the `ftrace_regs` data structure at the entry and exit. Note that
+        the instruction pointer of @fregs may be incorrect in entry handler and
+        exit handler, so you have to use @entry_ip and @ret_ip instead.

 @entry_data
         This is a local storage to share the data between entry and exit handlers.
```

Using quilt to fix git am errors

```
>$ quilt files
Documentation/trace/fprobe.rst
```

Using quilt to fix git am errors

```
>$ quilt files
Documentation/trace/fprobe.rst

>$ git add `quilt files`
```

Using quilt to fix git am errors

```
>$ quilt files
Documentation/trace/fprobe.rst

>$ git add `quilt files`

>$ git am --continue
Applying: Documentations: probes: Update fprobe document to use ftrace_regs
Applying: Documentation: tracing: Add a note about argument and retval access
```

More on quilt

- git is great!

More on quilt

- git is great!
- I make a lot of branches

More on quilt

- git is great!
- I make a lot of branches
- But too many branches

More on quilt

- git is great!
- I make a lot of branches
- But too many branches

```
>$ git branch | wc -l
688
```

git branch hoarder support group!

“Hello, my name is Steven, and I’m a git branch hoarder!”

More on quilt

- git is great!
- I make a lot of branches
- But too many branches
  - I work on lots of different projects
  - I'll save every version in its own branch

More on quilt

- git is great!
- I make a lot of branches
- But too many branches
  - I work on lots of different projects
  - I'll save every version in its own branch
- Hard to find what you looked for

More on quilt

- git is great!
- I make a lot of branches
- But too many branches
  - I work on lots of different projects
  - I'll save every version in its own branch
- Hard to find what you looked for
- Can at least see what I worked on last!

git-ls https://rostedt.org/code/git-ls

```
>$ git-ls | tail -20

9889f929b2eb 2023-07-17    trace/ftrace/mmap-ringbuffer-v3     try moving head page?
6e1eed2cb328 2023-07-19    trace/ftrace/eventfs-v10-works-with-files tracefs: Add show_events_dentries
4c414abcb5d9 2023-07-20    trace/ftrace/eventfs-v11-remove-eventfs_file eventfs: Remove eventfs_file and just use eventfs_inode
f345260dd301 2023-07-20    trace/ftrace/eventfs-v12-event-inode-almost tracefs: Add show_events_dentries
5a984d1d7aed 2023-07-20    trace/ftrace/eventfs-v13            tracefs: Add show_events_dentries
800c7938a9c9 2023-07-21    trace/ftrace/eventfs-v14            tracefs: Add show_events_dentries
3179de292e31 2023-07-24    trace/ftrace/eventfs-v15-works-without-eventfs-file tracefs: Add show_events_dentries
a6357b158413 2023-07-24    trace/ftrace/eventfs-v16            test: ftrace: Fix kprobe test for eventfs
62932e1fabdd 2023-07-28    trace/ftrace/eventfs-v17            test: ftrace: Fix kprobe test for eventfs
cd9a3751ae34 2023-07-29    cpumask-filters                     tracing/filters: Document cpumask filtering
e85b3b6c61dd 2023-07-29    trace/ftrace/mmap-ringbuffer-v4     tracing: Allow user-space mapping of the ring-buffer
11cc8b595f4d 2023-07-30    trace/ftrace/mmap-ringbuffer        tracing: Allow user-space mapping of the ring-buffer
1bde7feb665d 2023-07-31    trace/ftrace/eventfs-v18            tracefs: Add show_events_dentries
0348f7295322 2023-07-31    trace/ftrace/eventfs                eventfs: Remove eventfs_file and just use eventfs_inode
ef229c4c0cfb 2023-08-22    fprobe-btf                          Documentation: tracing: Update fprobe event example with BTF field
18940dd831de 2023-08-22    trace/tools/core                    rtla: Fix uninitialized variable found
58df6975ff88 2023-09-14    trace/show_events_dentries          tracing/selftests: Update kprobe args char/string to match new
functions
63362217fbaa 2023-09-14    trace/deadline-server               sched/fair: Fair server interface
799aecf7fcb3 2023-09-20    trace/ftrace/core                   squash this
e9c84be13c59 2023-09-23    trace/ftrace/urgent                 Documentation: tracing: Add a note about argument and retval
access
```

More on quilt

- For small changes and crazy POC work I use quilt

More on quilt

- For small changes and crazy POC work I use quilt
- Why?

More on quilt

- For small changes and crazy POC work I use quilt
- Why?
  - #1 reason is that it is easy to find again

More on quilt

- For small changes and crazy POC work I use quilt
- Why?
  - #1 reason is that it is easy to find again
  - Finding something you did hidden in a git branch is hard

More on quilt

- For small changes and crazy POC work I use quilt
- Why?
  - #1 reason is that it is easy to find again
  - Finding something you did hidden in a git branch is hard
  - I want to easily search for it

More on quilt

- For small changes and crazy POC work I use quilt
- Why?
  - #1 reason is that it is easy to find again
  - Finding something you did hidden in a git branch is hard
  - I want to easily search for it
- quilt saves its files in ./patches directory

More on quilt

- For small changes and crazy POC work I use quilt
- Why?
  - #1 reason is that it is easy to find again
  - Finding something you did hidden in a git branch is hard
  - I want to easily search for it
- quilt saves its files in ./patches directory

```
>$ ls patches/*.patch | wc -l
926
```

More on quilt

- For small changes and crazy POC work I use quilt
- Why?
  - #1 reason is that it is easy to find again
  - Finding something you did hidden in a git branch is hard
  - I want to easily search for it
- quilt saves its files in ./patches directory

```
>$ ls patches/*.patch | wc -l
926
```

```
>$ grep -l early_printk patches/*.patch
patches/early-printk-sync-v3.11.patch
patches/early-printk-sync-v3.12.patch
patches/early-printk-sync-v4.1.patch
patches/fix-4.2-rc1.patch
patches/nmi-idt-with-nmi.patch
patches/peterz-early-printk.patch
```

More on quilt

- For small changes and crazy POC work I use quilt
- Why?
  - #1 reason is that it is easy to find again
  - Finding something you did hidden in a git branch is hard
  - I want to easily search for it
- quilt saves its files in ./patches directory
- I use git to make the patch for me

More on quilt

- For small changes and crazy POC work I use quilt
- Why?
  - #1 reason is that it is easy to find again
  - Finding something you did hidden in a git branch is hard
  - I want to easily search for it
- quilt saves its files in ./patches directory
- I use git to make the patch for me

```
>$ git diff > /tmp/my-new-change.patch
```

More on quilt

- For small changes and crazy POC work I use quilt
- Why?
  - #1 reason is that it is easy to find again
  - Finding something you did hidden in a git branch is hard
  - I want to easily search for it
- quilt saves its files in ./patches directory
- I use git to make the patch for me

```
>$ git diff > /tmp/my-new-change.patch

>$ quilt import /tmp/my-new-change.patch
```

More on quilt

- For small changes and crazy POC work I use quilt
- Why?
  - #1 reason is that it is easy to find again
  - Finding something you did hidden in a git branch is hard
  - I want to easily search for it
- quilt saves its files in ./patches directory
- I use git to make the patch for me

```
>$ git diff > /tmp/my-new-change.patch

>$ quilt import /tmp/my-new-change.patch

>$ patch -R < /tmp/my-new-change.patch
```

More on quilt

- For small changes and crazy POC work I use quilt
- Why?
  - #1 reason is that it is easy to find again
  - Finding something you did hidden in a git branch is hard
  - I want to easily search for it
- quilt saves its files in ./patches directory
- I use git to make the patch for me
- tglx told me he uses git to maintain his ./patches directory!

Building and testing the kernel

- I do not use make & make modules & make install & make modules_install

Building and testing the kernel

- I do not use make & make modules & make install & make modules_install
- I only use ktest.pl
  - see tools/testing/ktest/ in the source tree

Building and testing the kernel

- I do not use make & make modules & make install & make modules_install
- I only use ktest.pl
  - see tools/testing/ktest/ in the source tree
- It uses a config to tell it what to do
  - Sample configs are in the examples directory

Building and testing the kernel

- I do not use make & make modules & make install & make modules_install
- I only use ktest.pl
  - see tools/testing/ktest/ in the source tree
- It uses a config to tell it what to do
  - Sample configs are in the examples directory
- I have a config file for each of my test machines (VMs and bare metal)

Building and testing the kernel

- I do not use make & make modules & make install & make modules_install
- I only use ktest.pl
  - see tools/testing/ktest/ in the source tree
- It uses a config to tell it what to do
  - Sample configs are in the examples directory
- I have a config file for each of my test machines (VMs and bare metal)
- I use libvirt for my VMs
  - Allows me to use virsh commands

```
# In another window: sudo virsh console devel-vm |
#                          tee /tmp/devel-vm | nc -kl localhost 4444
CONSOLE = nc localhost 4444

OUTPUT_DIRS = /work/build/nobackup
REBOOT_TYPE := grub2

INCLUDE include/default.conf
INCLUDE include/add-configs.conf
#INCLUDE include/bisect.conf

RUN_TYPE := boot
BUILD_NOCLEAN = 1

TEST_START IF ${RUN_TYPE} == boot && NOT DEFINED RUN_TEST
POST_TEST = echo -n "Build version: "; cat ${OUTPUT_DIR}/.version
TEST_TYPE = boot
#BUILD_TYPE = nobuild
#TEST_TYPE = build
BUILD_TYPE = oldconfig
#TEST_TYPE = test
#TEST_TYPE = install
#BUILD_TYPE = useconfig:/tmp/config-bad

DEFAULTS OVERRIDE IF ${USE_TEST_DIR}
BUILD_DIR = ${THIS_DIR}/linux-test.git
OUTPUT_DIR = ${THIS_DIR}/nobackup/${MACHINE}/test

DEFAULTS OVERRIDE
#MIN_CONFIG =
POST_INSTALL =
```

Running ktest.pl

```
>$ ./ktest.pl devel-vm.conf

[..]

Devel-vm login: Successful boot found: break after 1 second
kill child process 1829160
wait for child process 1829160 to exit
closing!

Build time:   1 minute 45 seconds
Install time: 6 seconds
Reboot time:  25 seconds



*******************************************
*******************************************
KTEST RESULT: TEST 1 SUCCESS!!!!   **
*******************************************
*******************************************
echo -n "Build version: "; cat /work/build/nobackup/devel-vm/.version ... [0 seconds] SUCCESS

    1 of 1 tests were successful


 See /work/build/nobackup/devel-vm/devel-vm.log for the record of results.
```

Testing

- I test with two VMs
  - One 64 bit and one 32 bit

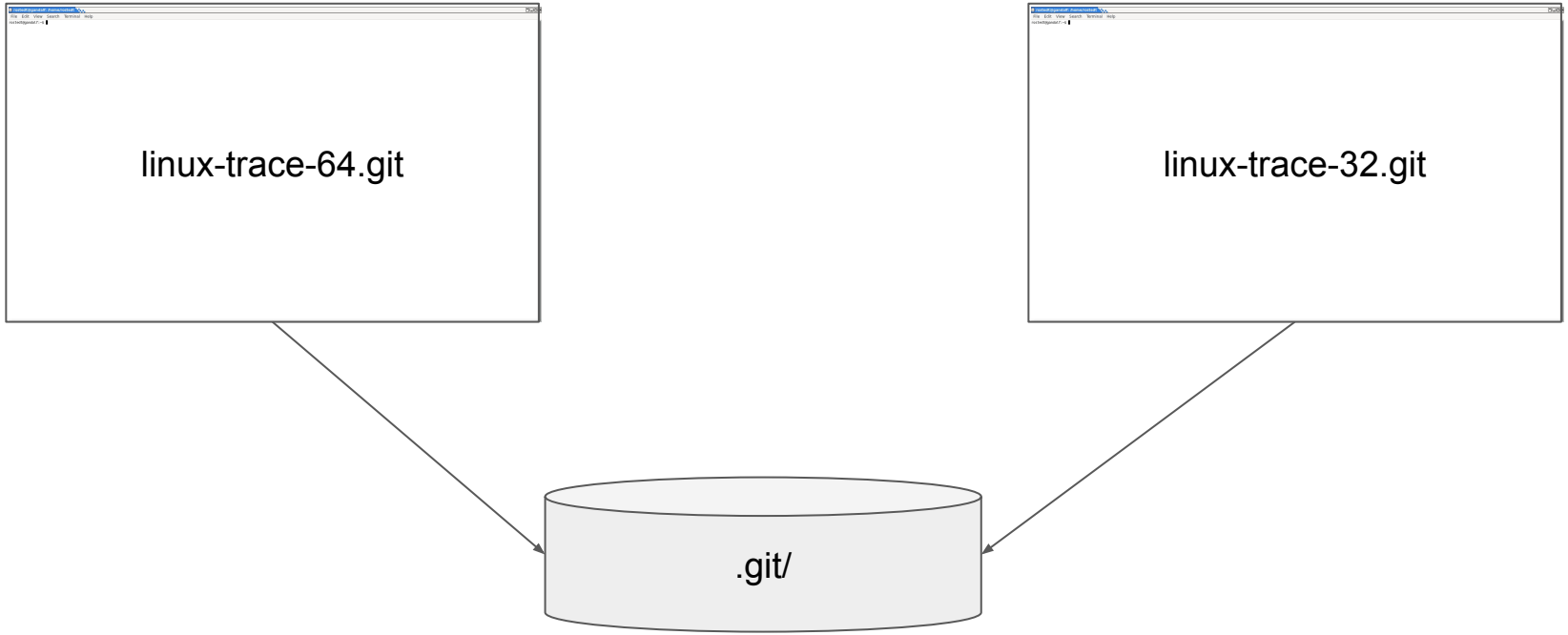
Testing

- I test with two VMs
  - One 64 bit and one 32 bit
- I use git worktree
  - One for testing 64 bit builds
  - One for testing 32 bit builds

Testing

- I test with two VMs
  - One 64 bit and one 32 bit
- I use git worktree
  - One for testing 64 bit builds
  - One for testing 32 bit builds
- I have two ktest configs, one for each VM
  - I run them in parallel
  - 32 bit has 10 tests
  - 64 bit has 35 tests

git worktrees



Tests are on github

https://github.com/rostedt/ftrace-ktests

- ktest.pl config files for both 64 and 32 bit machines

https://github.com/rostedt/ftrace-tests

- Tests on the machines that are run to test ftrace

Tests are on github

https://github.com/rostedt/ftrace-ktests

- ktest.pl config files for both 64 and 32 bit machines

https://github.com/rostedt/ftrace-tests

- Tests on the machines that are run to test ftrace

I do need to update them 😛

When the tests are finished

- The ktest configs are set to tag the commits
  - tracetest-tested-20230922-2100 for 64 bit
  - tracetest-32-tested-20230922-1824 for 32 bit

When the tests are finished

- The ktest configs are set to tag the commits
  - tracetest-tested-20230922-2100 for 64 bit
  - tracetest-32-tested-20230922-1824 for 32 bit
- Can use the tag to do diffs
  - and make sure not to accidentally test them again!

When the tests are finished

- The ktest configs are set to tag the commits
  - tracetest-tested-20230922-2100 for 64 bit
  - tracetest-32-tested-20230922-1824 for 32 bit
- Can use the tag to do diffs
  - and make sure not to accidentally test them again!
- Do a git pull from the server to the workstation

When the tests are finished

- The ktest configs are set to tag the commits
  - tracetest-tested-20230922-2100 for 64 bit
  - tracetest-32-tested-20230922-1824 for 32 bit
- Can use the tag to do diffs
  - and make sure not to accidentally test them again!
- Do a git pull from the server to the workstation
- Send out the for-next or for-linus patch series
  - This triggers the patchwork updates (state goes to “Queued”)

When the tests are finished

- The ktest configs are set to tag the commits
  - tracetest-tested-20230922-2100 for 64 bit
  - tracetest-32-tested-20230922-1824 for 32 bit
- Can use the tag to do diffs
  - and make sure not to accidentally test them again!
- Do a git pull from the server to the workstation
- Send out the for-next or for-linus patch series
  - This triggers the patchwork updates (state goes to “Queued”)
- I run my scripts: make-next or make-linus
  - Creates a quilt series file of all the commits to send out
  - And a prog file, I can update to include in my cover letter

Sending the patches to the lists

For fixes going to Linus:

```
>$ quilt mail --send --prefix 'for-linus][PATCH' --sender 'rostedt@goodmis.org' \
 --from 'Steven Rostedt <rostedt@goodmis.org>' --to 'linux-kernel@vger.kernel.org' \
 --cc 'Masami Hiramatsu <mhiramat@kernel.org>,Mark Rutland <mark.rutland@arm.com>,Andrew Morton
<akpm@linux-foundation.org>' --bcc 'rostedt@goodmis.org'
```

For going to linux-next:

```
>$ quilt mail --send --prefix 'for-next][PATCH' --sender 'rostedt@goodmis.org' \
 --from 'Steven Rostedt <rostedt@goodmis.org>' --to 'linux-kernel@vger.kernel.org' \
 --cc 'Masami Hiramatsu <mhiramat@kernel.org>,Mark Rutland <mark.rutland@arm.com>,Andrew Morton
<akpm@linux-foundation.org>' --bcc 'rostedt@goodmis.org'
```

Questions?