VAMPYR: Configurability-Aware Compile-Testing of Source Files

Stefan Hengelein Daniel Lohmann

stefan.hengelein@fau.de dl@cs.fau.de

System Software Group
Friedrich-Alexander University Erlangen-Nürnberg (FAU)

`https://cados.cs.fau.de`

LPC '14

supported by DFG

Kernel Patch Submission Checklist

- Before submitting patches kernel developers are expected to follow:

  ...

  8 Has been carefully reviewed with respect to relevant Kconfig combinations. This is very hard to get right with testing – brainpower pays off here.

  ...

Kernel Patch Submission Checklist

- Before submitting patches kernel developers are expected to follow:

  ...

  8 Has been carefully reviewed with respect to relevant Kconfig combinations. This is very hard to get right with testing – **brainpower pays off here.**

  ...

why is that a problem?

Configurability-aware compile testing

- Compile-test BLOCK1 and BLOCK2

```
#ifdef CONFIG_A
    block1
#else
    block2
#endif
```

Configurability-aware compile testing

- Compile-test BLOCK1 and BLOCK2

```
#ifdef CONFIG_A
    block1
#else
    block2
#endif
```

- However, one .CONFIG cannot cover both blocks

Configurability-aware compile testing

- Compile-test BLOCK1 and BLOCK2

```
#ifdef CONFIG_A
    block1
#else
    block2
#endif
```

- However, one .CONFIG cannot cover both blocks
- Code is often compile-tested with one `allyesconfig`

Configurability-aware compile testing

- Compile-test BLOCK1 and BLOCK2

```
#ifdef CONFIG_A
    block1
#else
    block2
#endif
```

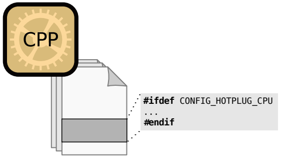
- However, one .CONFIG cannot cover both blocks

- Code is often compile-tested with **one** `allyesconfig`

Bugs are easily missed!

How hard is the problem?

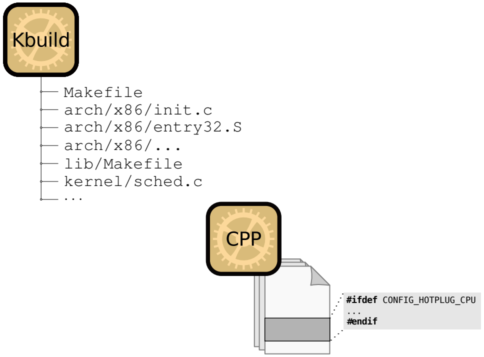
Configurability:



```
#ifdef CONFIG_HOTPLUG_CPU
...
#endif
```

How hard is the problem?

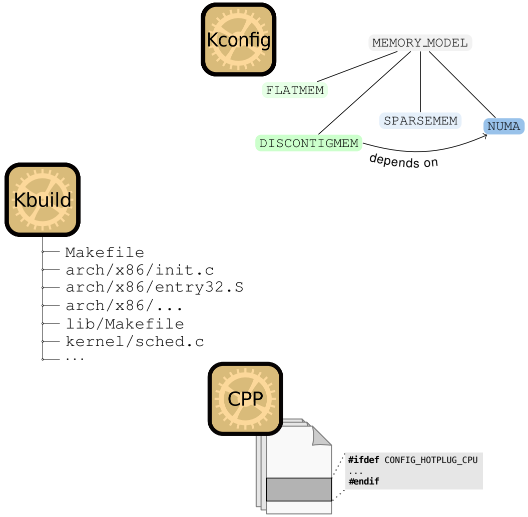
Configurability:



```
#ifdef CONFIG_HOTPLUG_CPU
...
#endif
```

How hard is the problem?

Configurability:

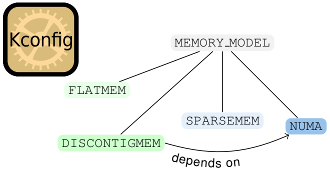


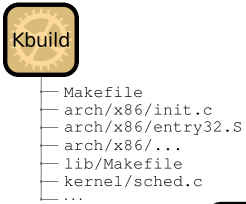
```
Makefile
arch/x86/init.c
arch/x86/entry32.S
arch/x86/...
lib/Makefile
kernel/sched.c
...
```

```
#ifdef CONFIG_HOTPLUG_CPU
...
#endif
```

How hard is the problem?

Configurability:

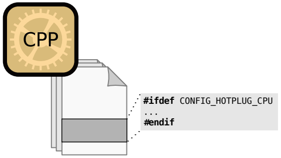




```
Makefile
arch/x86/init.c
arch/x86/entry32.S
arch/x86/...
lib/Makefile
kernel/sched.c
...
```

Still solvable by brainpower?

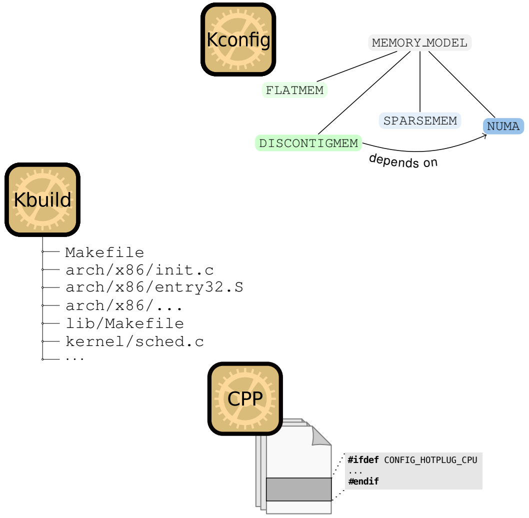
How much time does it take to get it *right*?



```
#ifdef CONFIG_HOTPLUG_CPU
...
#endif
```

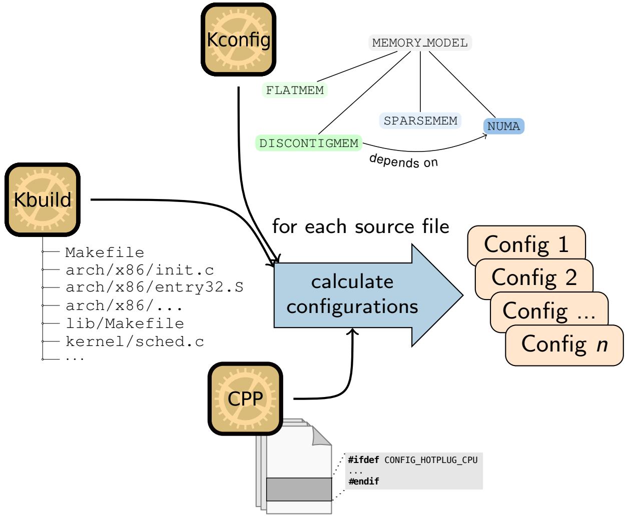
Configuration Coverage: The VAMPYR Approach

Idea: Create a maximizing set of configurations



Configuration Coverage: The VAMPYR Approach

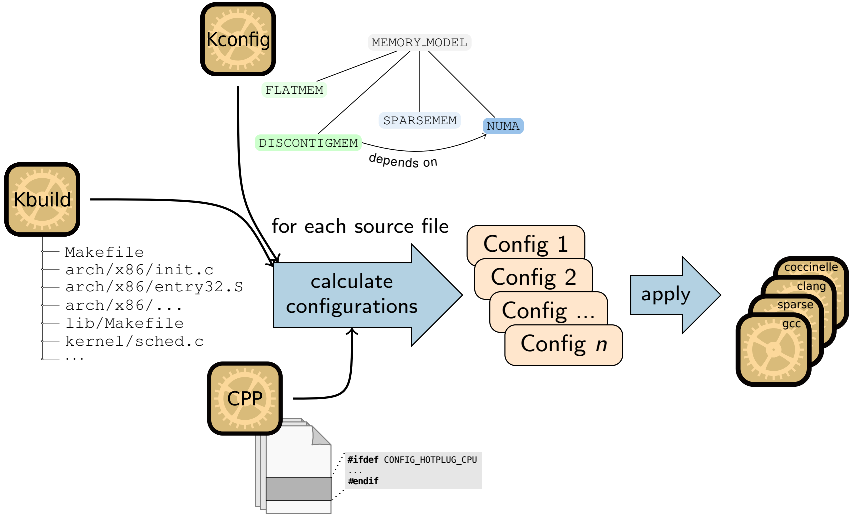
Idea: Create a maximizing set of configurations



```
#ifdef CONFIG_HOTPLUG_CPU
...
#endif
```

Configuration Coverage: The VAMPYR Approach

Idea: Create a maximizing set of configurations



Evaluation: A Configurability-Aware Tool Driver

Evaluated for Linux/`v3.2` [a]

- Number of found compiler warnings and errors increased significantly

Architecture	Increase in detected GCC warnings and errors
Linux/`x86`	176 → 202 (+15%)
Linux/`mips`	158 → 249 (+58%)

- Linux/`arm`: Analysis of Warnings and Errors not found with `allyesconfig`

Σ Less critical GCC messages	223 → 363	(+63%)
Σ Reported issues by VAMPYR	254 → 454	(+79%)
Σ Manually validated bugs	31 → 91	

[a] `https://www4.cs.fau.de/Publications/2014/tartler_14_usenix.pdf`

Evaluation: A Configurability-Aware Tool Driver

Evaluated for Linux/v3.2 [a]

- Number of found compiler warnings and errors increased significantly

Architecture	Increase in detected GCC warnings and errors
Linux/x86	176 → 202 (+15%)
Linux/mips	158 → 249 (+58%)

- Linux/arm: A … with allyesconfi…

Luckily:
The number of found warnings and errors is lower in Linux/v3.17

Σ Less critical GCC messages	223 → 363	(+63%)
Σ Reported issues by VAMPYR	254 → 454	(+79%)
Σ Manually validated bugs	31 → 91	

[a] https://www4.cs.fau.de/Publications/2014/tartler_14_usenix.pdf

Example 1 (v3.17 - MIPS)

```
===== Found 1 messages with gcc in arch/mips/ath79/mach-db120.c =====
... mach-db120.c:132: error: too many arguments to function 'db120_pci_init'
(in configs: arch/mips/ath79/mach-db120.c.config1)
========================================
```

```
#ifdef CONFIG_PCI
static void __init db120_pci_init(u8 *eeprom) { [...] }
#else
static void __init db120_pci_init(void) {}
#endif

static void __init db120_setup(void) {
    [...]
    db120_pci_init(art + DB120_PCIE_CALDATA_OFFSET);
}
```

Example 1 (v3.17 - MIPS)

```
===== Found 1 messages with gcc in arch/mips/ath79/mach-db120.c =====
... mach-db120.c:132: error: too many arguments to function ' db120_pci_init'
(in configs: arch/mips/ath79/mach-db120.c.config1)
=====================================================================
```

```
#ifdef CONFIG_PCI
static void __init db120_pci_init(u8 *eeprom) { [...] }
#else
static void __init db120_pci_init(void) {}
#endif

static void __init db120_setup(void) {
    [...]
    db120_pci_init(art + DB120_PCIE_CALDATA_OFFSET);
}
```

Conclusions

- intentional broken compilation?
- why not #error?
- or model configurability better

Example 2 (v3.17 - ARM)

```
==== Found 8 messages with gcc in arch/arm/mm/dma-mapping.c ====
... dma-mapping.c:1358: error: 'atomic_pool' undeclared (first use in this function)
(in configs: arch/arm/mm/dma-mapping.c.config1)
... dma-mapping.c:1369: error: implicit declaration of function '__in_atomic_pool'
(in configs: arch/arm/mm/dma-mapping.c.config1) ...
==================================================
```

```
#ifdef CONFIG_MMU
static struct gen_pool *atomic_pool;
static bool __in_atomic_pool([...]) {
        return addr_in_gen_pool(atomic_pool, [...]);
}
#endif

static struct page **__atomic_get_pages(void *addr) {
    phys = gen_pool_virt_to_phys(atomic_pool, [...]);
}
static struct page **__iommu_get_pages([...]) {
    if (__in_atomic_pool(cpu_addr, PAGE_SIZE))
        return __atomic_get_pages(cpu_addr);
}
```

Example 2 (v3.17 - ARM)

```
==== Found 8 messages with gcc in arch/arm/mm/dma-mapping.c ====
... dma-mapping.c:1358: error: 'atomic_pool' undeclared (first use in this function)
(in configs: arch/arm/mm/dma-mapping.c.config1)
... dma-mapping.c:1369: error: implicit declaration of function '__in_atomic_pool'
(in configs: arch/arm/mm/dma-mapping.c.config1) ...
=================================================================
```

declared inside #ifdef

```
#ifdef CONFIG_MMU
static struct gen_pool *atomic_pool;
static bool __in_atomic_pool([...]) {
        return addr_in_gen_pool(atomic_pool, [...]);
}
#endif
```

used outside #ifdef

```
static struct page **__atomic_get_pages(void *addr) {
    phys = gen_pool_virt_to_phys(atomic_pool, [...]);
}
static struct page **__iommu_get_pages([...]) {
    if (__in_atomic_pool(cpu_addr, PAGE_SIZE))
        return __atomic_get_pages(cpu_addr);
}
```

Example 2 (v3.17 - ARM)

```
==== Found 8 messages with gcc in arch/arm/mm/dma-mapping.c ====
... dma-mapping.c:1358: error: ' atomic_pool' undeclared (first use in this function)
(in configs: arch/arm/mm/dma-mapping.c.config1)
... dma-mapping.c:1369: error: implicit declaration of function ' __in_atomic_pool'
(in configs: arch/arm/mm/dma-mapping.c.config1) ...
=====================================
```

```
#ifdef CONFIG_MMU
static struct gen_pool *atomic_pool;
static bool __in_atomic_pool([...]) {
        return addr_in_gen_pool(atomic_pool, [...]);
}
#endif

static struct page **__atomic_get_pages(void *addr) {
    phys = gen_pool_virt_to_phys(atomic_pool, [...]);
}
static struct page **__iommu_get_pages([...]) {
    if (__in_atomic_pool(cpu_addr, PAGE_SIZE))
        return __atomic_get_pages(cpu_addr);
}
```

Example 2 (v3.17 - ARM)

```
==== Found 8 messages with gcc in arch/arm/mm/dma-mapping.c ====
... dma-mapping.c:1358: error: ' atomic_pool' undeclared (first use in this function)
(in configs: arch/arm/mm/dma-mapping.c.config1)
... dma-mapping.c:1369: error: implicit declaration of function ' __in_atomic_pool'
(in configs: arch/arm/mm/dma-mapping.c.config1) ...
================================================
```

```
#ifdef CONFIG_MMU
static struct gen_pool *atomic_pool;
static bool __in_atomic_pool([...]) {
        return addr_in_gen_pool(atomic_pool, [...]);
}
#endif

static struct page **__atomic_get_pages(void *addr) {
    phys = gen_pool_virt_to_phys(atomic_pool, [...]);
}
static struct page **__iommu_get_pages([...]) {
    if (__in_atomic_pool(cpu_addr, PAGE_SIZE))
        return __atomic_get_pages(cpu_addr);
}
```

defined inside #ifdef

used outside #ifdef

Conclusions

- #ifdef around 'uses' missing or declaration should have been unconditional

Example 3 (v3.17 - MIPS)

```
==== Found 1 messages with gcc in arch/mips/pmcs-msp71xx/msp_irq_cic.c ====
... msp_irq_cic.c:134: error: 'irq' undeclared (first use in this function)
(in configs: arch/mips/pmcs-msp71xx/msp_irq_cic.c.config0)
=====================================================
```

```
#ifdef CONFIG_MIPS_MT_SMP
static int msp_cic_irq_set_affinity(struct irq_data *d,
    [...]) {
    unsigned long imask = (1 << (irq - MSP_CIC_INTBASE))
        ;
    [...]
    BUG_ON(irq) == MSP_INT_VPE0_TIMER || irq ==
        MSP_INT_VPE1_TIMER);
}
#endif
```

Example 3 (v3.17 - MIPS)

```
==== Found 1 messages with gcc in arch/mips/pmcs-msp71xx/msp_irq_cic.c ====
... msp_irq_cic.c:134: error: 'irq' undeclared (first use in this function)
(in configs: arch/mips/pmcs-msp71xx/msp_irq_cic.c.config0)
========================================================
```

```
#ifdef CONFIG_MIPS_MT_SMP
static int msp_cic_irq_set_affinity(struct irq_data *d,
    [...]) {
    unsigned long imask = (1 << (irq - MSP_CIC_INTBASE))
        ;
    [...]
    BUG_ON(irq) == MSP_INT_VPE0_TIMER || irq ==
        MSP_INT_VPE1_TIMER);
}
#endif
```

Conclusions

- obviously never compiled
- should have been **d->irq**

Compile Testing

> *“Major problem I see is that many architecture maintainers don’t seem to care about* MAKE ALLMODCONFIG *and/or* MAKE ALLYESCONFIG, *meaning there is no simple means to at least compile-test all code that _can_ be enabled for a given architecture. And don’t even mention* MAKE RANDCONFIG. *”*
>
> Guenter Roeck

Compile Testing

> *“Major problem I see is that many architecture maintainers don’t seem to care about* MAKE ALLMOD-
> CONFI *… there*
> *is no … code*
> *that … And*
> *don’t even mention* MAKE RANDCONFIG.*”*
>
> Guenter Roeck

Integrate VAMPYR into your development workflow! replace brainpower by tools!

Compile Testing

“Major problem I see is that many architecture maintainers don’t seem to care about MAKE ALLMOD-CONF*[illegible] there is no [illegible] code that [illegible] And don’t even mention* MAKE RANDCONFIG. *”*

Guenter Roeck

Integrate VAMPYR into
your development workflow!
replace brainpower by tools!

Summary and Conclusions

- VAMPYR is a tool to help developers to compile-test their source code
- The tool is available under **GPLv3**!
- VAMPYR has been applied to other configurable system software: `busybox`, `L4/Fiasco`
- Integration into UNDERTAKER–CHECKPATCH is on the way

Interested?

- Download and try the tool:

  `https://undertaker.cs.fau.de`

- More information and papers on the project's website:

  `https://cados.cs.fau.de`

- Questions? Contact me directly ...

  stefan.hengelein@fau.de

- ... or write to our mailing list!

  cados-dev@lists.cs.fau.de

Backup Slides Start

Evaluation: Analysis of Warnings and Errors

- VAMPYR reveals issues not covered by `allyesconfig`:

	Compiler Diagnostics	
Less critical GCC messages		
Σ Less critical messages	223 → 363	+140
Manually validated bugs		
Undeclared types/identifiers	4 → 46	+42
Access to possibly uninitialized data	20 → 22	+2
Out of bounds array accesses	7 → 13	+4
Bad pointer casts	0 → 8	
Format string warnings	0 → 1	
Integer overflows	0 → 1	
Σ Bugs found	**31 → 91**	+60
Σ Reported issues by VAMPYR	254 → 454	+200

- Seven patches were submitted and accepted by Linux maintainers

Configuration Coverage

- Common approach: use a single configuration (i.e. `allyesconfig`)
- Configuration Coverage is the percentage of code that is covered by all tested configurations
- This is insufficient because for `allyesconfig`:

Configuration Coverage on	v3.2	v3.17-rc7
Linux/`x86`	78.6%	77%
Linux/`arm`	59.7%	69.5%
Linux/`mips`	54.6%	52.4%

Evaluation: Setup and Runtime Requirements

- Application of VAMPYR on all 24 Linux Architectures of Linux v3.2
- Used Static Checker: GCC: 4.7
- On average, 1.2 compiler invocations per file

$\rightsquigarrow$ Overhead $\sim$ 20%

- Runtime on a Standard Intel Quad-Core Workstation:
  - Incremental analysis of an individual file: $<$ 1 minute
  - Generation of 11470 (on Linux/x86) partial configurations in $\sim$ 4 minutes
  - Analysis of a full architecture: $\sim$ 2 hours
  - Majority of time is spent with activating KCONFIG configurations

Why not 100 percent Configuration coverage?

- Bugs in KCONFIG descriptions in the Linux kernel can cause incorrect expansions of partial configurations.
- Imperfect model extractors can also lower the Configuration Coverage

Higher Coverage Criteria

- VAMPYR: **statement coverage** ($\sim$ 20% overhead)
- Possibly achievable: **decision coverage**: ($\sim$ 29% overhead)
- Expensive: **path coverage**: (?? overhead)
- Further research: Pairwise testing (cf. related work)