LLVMLinux: Patch status

Presented by:
Behan Webster
(LLVMLinux project lead)

Presentation Date: 2014.10.17
The LLVMLinux Project Goals

- Fully build the Linux kernel for multiple architectures, using the Clang/LLVM toolchain
- Discover LLVM/Kernel issues early and find fixes quickly across both communities
- Upstream patches to the Linux Kernel and LLVM projects
- Bring together like-minded developers
- Enable the kernel community to do more in depth analysis of the kernel code
A mainline kernel tree with all LLVMLinux patches applied on top is now available:

- [git://git.linuxfoundation.org/llvmlinux/kernel.git](git://git.linuxfoundation.org/llvmlinux/kernel.git)

Dated llvmlinux branches

- [remotes/origin/llvmlinux-2014.10.15](remotes/origin/llvmlinux-2014.10.15)

The master branch is rebased regularly

Also a part of linux-next and 0-day kbuild-robot
LLVM/Linux Project Status

- LLVM/clang:
  - All LLVM/Linux patches for LLVM are Upstream
  - Newer LLVM patches to support the Linux kernel are mostly being added by upstream maintainers
  - Named register support for sp in clang-3.6-svn
  - A number of ASM related issues
## LLVMLinux Kernel Patches

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Number of patches</th>
<th>Submitted</th>
<th>Acked</th>
<th>Accepted Upstream</th>
</tr>
</thead>
<tbody>
<tr>
<td>all</td>
<td>27</td>
<td>8</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>arm</td>
<td>17</td>
<td>13</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>aarch64</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>x86_64</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TOTALS</td>
<td>49</td>
<td>21</td>
<td>12</td>
<td>5</td>
</tr>
</tbody>
</table>
Integrated Assembly Status

- David Woodhouse added .code16 support for X86 ASM
- Renato Golin, Vicinius Tinti, Saleem Abdulrasool and Stepan Dyatkovskiy are working on fixing IA issues in clang to support the Linux ARM kernel code (and ultimately AARCH64)
- For now we disable the IA and use GNU as instead
Different option passing

• gcc passes -march to GNU as
• clang doesn't... (Bug submitted PR)

-CFLAGS_aes-ce-cipher.o += -march=armv8-a+crypto
+CFLAGS_aes-ce-cipher.o += -march=armv8-a+crypto -Wa,-march=armv8-a+crypto
extern inline: Different for gnu89 and gnu99

- GNU89/GNU90 (used by gcc)
  - Function will be inlined where it is used
  - No function definition is emitted
  - A non-inlined function may also be provided
- GNU99/C99 (used by clang)
  - Function will be inlined where it is used
  - An external function is emitted
  - No other function of the same name may be provided.
- Solution? Use “static inline” instead.
- Only still an issue for ARM support for ftrace (submitted)
Attribute Order

- gcc is less picky about placement of `__attribute__(())`
- clang requires it at the end of the type or variable

```c
-struct __read_mostly va_alignment va_align = {
+struct va_alignment __read_mostly va_align = {
```
ARM percpu patch

- One of the uses of Named Registers in the ARM code is due to a deficiency in gcc.
- The new code which works with gcc fails in clang.
- Solution, provide routines for both, and choose at compile time.
- Gcc:
  ```
  asm("mrc p15, 0, %0, c13, c0, 4" : "=r" (off) : "Q" (*sp));
  ```
- Clang:
  ```
  asm("mrc p15, 0, %0, c13, c0, 4" : "=r" (off) : : "memory");
  ```
Section Mismatch Issues (MergedGlobals)

- By default clang merges globals with internal linkage into one: MergedGlobals
- Allows globals to be addressed using offsets from a base pointer
- Can reduce the number of registers used
- Modpost uses symbol names to look for section mismatches
- MergedGlobals breaks modpost (false positive section mismatches)
- Current solution: use -mno-global-merge to stop global merging
- Updates to modpost may allow this optimization to be enabled again
ARM eabi support

- Clang emits code which uses the “aeabi” ARM calls which are implemented in compiler-rt (equivalent to libgcc)
- Compiler-rt doesn't easily cross compile yet...

```c
void __aeabi_memcpy(void *dest, const void *src, size_t n)
void __aeabi_memmove(void *dest, const void *src, size_t n)
void __aeabi_memset(void *s, size_t n, int c)
```
Variable Length Arrays In Structs

- VLAIS isn't supported by Clang (undocumented gcc extension)

```c
char vla[n];           /* Supported, C99/C11 */

struct {
    char flexible_member[]; /* Supported, C99/C11 */
} struct_with_flexible_member;

struct {
    char vlais[n];         /* Explicitly not allowed by C99/C11 */
} variable_length_array_in_struct;
```

- VLAIS is used in the Linux kernel in a number of places, spreading mostly through reusing patterns from data structures found in crypto
VLAIS Removal Example

- struct {
-       struct shash_desc shash;
-       char ctx[crypto_shash_descsize(tfm)];
- } desc;
+
+ char desc[sizeof(struct shash_desc) + crypto_shash_descsize(tfm)] CRYPTO_MINALIGN_ATTR;
+
+ struct shash_desc *shash = (struct shash_desc *)desc;

  unsigned int i;

- desc.shash.tfm = tfm;
+ shash->tfm = tfm;  
  
(from crypto/hmac.c)
VLAIS Removal Example

- struct {
-     struct shash_desc shash;
-     char ctx[crypto_shash_descsize(tfm)];
- } desc;
+ SHASH_DESC_ON_STACK(shash, tfm);

- desc.shash.tfm = tfm;
+ shash->tfm = tfm;
`#define ARCH_KMALLOC_MINALIGN __alignof__(unsigned long long)`

`#define CRYPTO_MINALIGN ARCH_KMALLOC_MINALIGN`

`#define CRYPTO_MINALIGN_ATTR __attribute__((__aligned__(CRYPTO_MINALIGN)))`

```c
struct shash_desc {
    struct crypto_shash *tfm;
    u32 flags;
    void *__ctx[] CRYPTO_MINALIGN_ATTR;
};
```
Bloat-o-meter gcc vs clang

<table>
<thead>
<tr>
<th>text</th>
<th>data</th>
<th>bss</th>
<th>dec</th>
<th>hex</th>
<th>filename</th>
</tr>
</thead>
<tbody>
<tr>
<td>5451700</td>
<td>230636</td>
<td>153632</td>
<td>5835968</td>
<td>590cc0</td>
<td>build/targets/vexpress/build/kernel-gcc/vmlinux</td>
</tr>
<tr>
<td>6178530</td>
<td>229100</td>
<td>154058</td>
<td>6561688</td>
<td>641f98</td>
<td>build/targets/vexpress/build/kernel-clang/vmlinux</td>
</tr>
</tbody>
</table>

Bloat-o-meter:
add/remove: 1781/996 grow/shrink: 8172/4922 up/down: 585929/-413928 (172001)

<table>
<thead>
<tr>
<th>function</th>
<th>old</th>
<th>new</th>
<th>delta</th>
</tr>
</thead>
<tbody>
<tr>
<td>__blockdev_direct_IO</td>
<td>5364</td>
<td>15052</td>
<td>+9688</td>
</tr>
<tr>
<td>cache_slow_downcall.write_buf</td>
<td>-</td>
<td>8192</td>
<td>+8192</td>
</tr>
<tr>
<td>do_con_write</td>
<td>-</td>
<td>7572</td>
<td>+7572</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>do_write_buffer</td>
<td>8008</td>
<td>-</td>
<td>-8008</td>
</tr>
<tr>
<td>write_buf</td>
<td>8192</td>
<td>-</td>
<td>-8192</td>
</tr>
<tr>
<td><strong>func</strong></td>
<td>11219</td>
<td>-</td>
<td>-11219</td>
</tr>
</tbody>
</table>
Contribute to the LLVMLinux Project

- **Project wiki page**
  - [http://llvm.linuxfoundation.org](http://llvm.linuxfoundation.org)

- **Project Mailing List**
  - [http://lists.linuxfoundation.org/mailman/listinfo/llvmlinux](http://lists.linuxfoundation.org/mailman/listinfo/llvmlinux)
  - [http://lists.linuxfoundation.org/pipermail/llvmlinux/](http://lists.linuxfoundation.org/pipermail/llvmlinux/)

- **IRC Channel**
  - #llvmlinux on OFTC

- **LLVMLinux Community on Google Plus**