PAHOLE
&
DATA-TYPE PROFILING UPDATE

Arnaldo Carvalho de Melo
acme@kernel.org
Red Hat
LSF/MM+BPF, Salt Lake City, 2024

PAHOLE

- --btf_features
- Reproducible builds
- DECL_TAG for kfuncs
- Distilled base BTF

--btf_features

- Simplify scripts/Makefile.btf
- Older pahole ignores unknown features
- --btf_features_strict while developing
- Check one last time for pahole version
- v1.26

scripts/Makefile.btf

```
# pahole 1.18 through 1.21 can't handle zero-sized per-CPU vars
ifeq ($(call test-le, $(pahole-ver), 121),y)
pahole-flags-$(call test-ge, $(pahole-ver), 118) += --skip_encoding_btf_vars
endif

pahole-flags-$(call test-ge, $(pahole-ver), 121) += --btf_gen_floats

pahole-flags-$(call test-ge, $(pahole-ver), 122) += -j

pahole-flags-$(CONFIG_PAHOLE_HAS_LANG_EXCLUDE)   += --lang_exclude=rust

pahole-flags-$(call test-ge, $(pahole-ver), 125) += --skip_encoding_btf_inconsistent_proto --btf_gen_optimized

export PAHOLE_FLAGS := $(pahole-flags-y)
```

With --btf_features

```
# Switch to using --btf_features for v1.26 and later.
pahole-flags-$(call test-ge, $(pahole-ver), 126) = -j --btf_features=encode_force,var,float,enum64,
                                                                     decl_tag,type_tag,optimized_func,
                                                                     consistent_func
```

REPRODUCIBLE BUILD

- kernel patch disabling parallel BTF encoding
- Need: same BTF generated for a given vmlinux
- Keep parallel DWARF loading
- Encode BTF in the same DWARF CU order
- tests/reproducible_build.sh

PERFORMANCE DIFF

- Minimal
- Less then 100ms reproducible/non-reproducible
- vmlinux BTF generation
- https://lore.kernel.org/all/82928441-d185-4165-85ff-425350953e80@oracle.com

DECL_TAG for kfuncs

- Stores BTF_KIND_DECL_TAG for kfuncs
- From .BTF_ids vmlinux ELF section
- Enables tools to find about kfuncs
- --btf_features=decl_tag,decl_tag_kfuncs

Encoding

```
$ time pahole -j --btf_features=decl_tag,decl_tag_kfuncs \
                 --btf_encode_detached=vmlinux.btf.decl_tag,decl_tag_kfuncs \
                 vmlinux-v6.9.0-rc7
real    0m5.938s
user    0m32.050s
sys     0m2.075s
```

Enumerating

```
$ bpftool btf dump file vmlinux.btf.decl_tag,decl_tag_kfuncs |
        grep -w DECL_TAG | head -5
[135450] DECL_TAG 'bpf_kfunc' type_id=94151 component_idx=-1
[135451] DECL_TAG 'bpf_kfunc' type_id=94146 component_idx=-1
[135452] DECL_TAG 'bpf_kfunc' type_id=74311 component_idx=-1
[135453] DECL_TAG 'bpf_kfunc' type_id=74309 component_idx=-1
[135454] DECL_TAG 'bpf_kfunc' type_id=74307 component_idx=-1
```

Enumerating

```
$ bpftool btf dump file vmlinux.btf.decl_tag,decl_tag_kfuncs |
        grep -w DECL_TAG | head -5
[135450] DECL_TAG 'bpf_kfunc' type_id=94151 component_idx=-1
[135451] DECL_TAG 'bpf_kfunc' type_id=94146 component_idx=-1
[135452] DECL_TAG 'bpf_kfunc' type_id=74311 component_idx=-1
[135453] DECL_TAG 'bpf_kfunc' type_id=74309 component_idx=-1
[135454] DECL_TAG 'bpf_kfunc' type_id=74307 component_idx=-1

$ bpftool btf dump file vmlinux.btf.decl_tag,decl_tag_kfuncs | grep
116
$
```

Dumping one

```
$ bpftool btf dump file vmlinux.btf.decl_tag,decl_tag_kfuncs | grep -w 94151
[94151] FUNC 'cgroup_rstat_updated' type_id=94150 linkage=static
[135450] DECL_TAG 'bpf_kfunc' type_id=94151 component_idx=-1
```

Dumping one

```
$ bpftool btf dump file vmlinux.btf.decl_tag,decl_tag_kfuncs | grep -w 94151
[94151] FUNC 'cgroup_rstat_updated' type_id=94150 linkage=static
[135450] DECL_TAG 'bpf_kfunc' type_id=94151 component_idx=-1

$ bpftool btf dump file vmlinux.btf.decl_tag,decl_tag_kfuncs | grep -w 94150 -A2
[94150] FUNC_PROTO '(anon)' ret_type_id=0 vlen=2
        'cgrp' type_id=744
        'cpu' type_id=12
[94151] FUNC 'cgroup_rstat_updated' type_id=94150 linkage=static
$
```

Dumping one

```
$ bpftool btf dump file vmlinux.btf.decl_tag,decl_tag_kfuncs | grep -w 94151
[94151] FUNC 'cgroup_rstat_updated' type_id=94150 linkage=static
[135450] DECL_TAG 'bpf_kfunc' type_id=94151 component_idx=-1

$ bpftool btf dump file vmlinux.btf.decl_tag,decl_tag_kfuncs | grep -w 94150 -A2
[94150] FUNC_PROTO '(anon)' ret_type_id=0 vlen=2
        'cgrp' type_id=744
        'cpu' type_id=12
[94151] FUNC 'cgroup_rstat_updated' type_id=94150 linkage=static
$

$ git grep '__bpf_kfunc.* cgroup_rstat_updated'
kernel/cgroup/rstat.c:__bpf_kfunc void cgroup_rstat_updated(struct cgroup *cgrp, int cpu)
$
```

While coming here

```
$ pfunct --prototypes -F btf vmlinux.btf.decl_tag,decl_tag_kfuncs | grep 'bpf_kfunc.*cgroup_rstat'
bpf_kfunc void cgroup_rstat_updated(struct cgroup * cgrp, int cpu);
bpf_kfunc void cgroup_rstat_flush(struct cgroup * cgrp);
```

While coming here

```
$ pfunct --prototypes -F btf vmlinux.btf.decl_tag,decl_tag_kfuncs | grep 'bpf_kfunc.*cgroup_rstat'
bpf_kfunc void cgroup_rstat_updated(struct cgroup * cgrp, int cpu);
bpf_kfunc void cgroup_rstat_flush(struct cgroup * cgrp);

$ pfunct --prototypes -F btf vmlinux.btf.decl_tag,decl_tag_kfuncs | grep ^bpf_kfunc | head -5
bpf_kfunc void cubictcp_init(struct sock * sk);
bpf_kfunc void cubictcp_cwnd_event(struct sock * sk, enum tcp_ca_event event);
bpf_kfunc void cubictcp_cong_avoid(struct sock * sk, u32 ack, u32 acked);
bpf_kfunc u32 cubictcp_recalc_ssthresh(struct sock * sk);
bpf_kfunc void cubictcp_state(struct sock * sk, u8 new_state);
$
```

RESILIENT SPLIT BTF

- Out of tree modules
- BTF ID drift when base kernel gets rebuilt
- .BTF.base ELF section in addition to .BTF
- split BTF in modules point to the .BTF.base
- That gets relocated if needed at load time
- Relative to /sys/kernel/btf/vmlinux
- Changes to bpftool, pahole, libbpf, the kernel
- Not yet merged

pahole todo

- Support kfunc decls in pfunct
- parallel reproducible encoding of BTF
- Further testing and merge of resilient split BTF

DATA-TYPE PROFILING

- Recap:
- perf mem
- perf c2c

PERF MEM

- PEBS on Intel
- mem-loads and mem-stores
- Data addresses
- Cache hierarchy
- record/report

RECORD

```
# echo 1 > /proc/sys/vm/drop_caches
# perf mem record find / > /dev/null
[ perf record: Woken up 1 times to write data ]
[ perf record: Captured and wrote 0.061 MB perf.data (26 samples) ]
#
```

EVENTS RECORDED

```
# perf evlist
cpu_atom/mem-loads,ldlat=30/P
cpu_atom/mem-stores/P
dummy:u
#
```

HYBRID SYSTEMS

- Missing cpu_core/ events
- Intel specific details
- Needs polishing
- To avoid boilerplate
- Use 'perf record' directly

BOILERPLATE

```
# taskset -c 0 \
  perf record --weight --data \
              --event '{cpu_core/mem-loads-aux/,cpu_core/mem-loads,ldlat=30/P}:S' \
              --event cpu_core/mem-stores/ find / > /dev/null
[ perf record: Woken up 20 times to write data ]
[ perf record: Captured and wrote 5.138 MB perf.data (79628 samples) ]
#
```

FINALLY

```
# perf evlist
cpu_core/mem-loads-aux/
cpu_core/mem-loads,ldlat=30/P
cpu_core/mem-stores/
dummy:u
#
```

REPORT

```
# perf mem report
# Total Lost Samples: 0
#
# Samples: 25K of event 'cpu_core/mem-loads-aux/'
# Total weight : 1123282
# Sort order   : local_weight,mem,sym,dso,symbol_daddr,dso_daddr,snoop,tlb,locked,blocked,local_ins_lat,local_p_
#
# Overhead Samples  LocalWeight Mem access  Symbol                       Shared Obj  Data Symbol
# ........ .......  ........... ........... ...........................  .......... ......................
#
     0.50%       1  5635        RAM hit     [k] btrfs_bin_search         [kernel]   [k] 0xffff90b3b9fe0a31
     0.22%       1  2504        RAM hit     [k] rb_next                  [kernel]   [k] 0xffff90af31bfcda8
     0.13%       1  1472        LFB/MAB hit [k] mutex_lock               [kernel]   [k] 0xffff90adca8c1d18
     0.13%       1  1432        LFB/MAB hit [k] btrfs_get_delayed_node   [kernel]   [k] 0xffff90b4c9a17158
     0.12%       1  1376        LFB/MAB hit [k] generic_fillattr         [kernel]   [k] 0xffff90b422422032
SNIP
     0.02%       1  220         L3 hit      [k] ktime_get_update_offsets_now [kernel] [k] tk_core+0xc0
SNIP
     0.02%       1  216         LFB/MAB hit [k] update_vsyscall          [kernel]   [k] shadow_timekeeper+0x40
SNIP
     0.02%       1  208         LFB/MAB hit [k] _raw_spin_lock           [kernel]   [k] jiffies_lock+0x0
```

--mem-mode --sort

```
# perf report --stdio --mem-mode --sort mem
# Samples: 26K of event 'cpu_core/mem-loads,ldlat=30/P'
# Total weight : 1135614
# Sort order   : mem
#
# Overhead  Memory access
# ........  ............
#
    62.32%  LFB/MAB hit
    24.22%  RAM hit
    10.28%  L1 hit
     2.40%  L3 hit
     0.78%  L2 hit
```

kernel functions doing mem loads

```
# perf report --dso '[kernel.kallsyms]' --stdio \
              --mem-mode --sort sym,ins_lat
# Overhead  Symbol                          INSTR Latency
# ........  ..............................  ..............
#
     0.50%  [k] btrfs_bin_search           5637
     0.22%  [k] rb_next                    2507
     0.18%  [k] folio_mark_accessed         419
     0.18%  [k] __d_lookup                  405
     0.17%  [k] __d_lookup_rcu              389
     0.14%  [k] down_read                    41
     0.14%  [k] __d_lookup_rcu              390
     0.13%  [k] mutex_lock                 1475
     0.13%  [k] mutex_lock                  487
     0.13%  [k] btrfs_get_delayed_node     1441
     0.12%  [k] generic_fillattr            703
     0.12%  [k] generic_fillattr           1378
     0.12%  [k] folio_mark_accessed        1371
     0.12%  [k] _raw_spin_lock               33
     0.12%  [k] btrfs_get_delayed_node      444
     0.11%  [k] dcache_readdir             1283
     0.11%  [k] __d_lookup_rcu              431
     0.11%  [k] folio_mark_accessed         640
```

#

PERF C2C

- record/report
- cacheline oriented
- shows cacheline offset
- source/line number
- Look at the source
- Figure out the data structure/member

HELPS

- Data-type profiling LWN article
- https://lwn.net/Articles/955709/

RESOLVING TYPES

- DWARF location expressions
- Parsing disassembled instructions
- Type info from DWARF

PERF ANNOTATE

- Disassembly
- Parsing objdump -dS output
- TUI navigation
- jumps, calls
- capstone for x86-64: faster
- Falls back to objdump when it fails
- Enable for PowerPC, etc
- Improving 'perf annotate'

REUSE IT FOR DATA-TYPE PROFILING

- parse more instructions
- mov, add, etc
- Not all right now
- PowerPC support being reviewed

MORE KEYS TO SORT

- type: struct, base type
- or type of memory (stack, etc)
- typeoff: offset, field name

REPORT EXAMPLE

```
# perf report --stdio -s type -i perf.data.mem.find
# Total Lost Samples: 0
#
# Samples: 25K of event 'cpu_core/mem-loads-aux/'
# Event count (approx.): 170070020
#
# Overhead  Data Type
# ........  .........
    18.34%  (stack operation)
    15.35%  struct btrfs_key
    10.83%  struct
     9.13%  (unknown)
     8.14%  int
     7.75%  unsigned int
     3.69%  long long unsigned int
     3.02%  (stack canary)
     2.62%  struct _ftsent
     2.61%  struct extent_buffer
     2.50%  struct extent_buffer*
     2.46%  struct __va_list_tag
     2.15%  struct inode
     2.12%  long unsigned int
     1.03%  struct btrfs_delayed_node
     0.86%  struct nameidata
     0.82%  struct dentry
     0.62%  struct mnt_idmap*
     0.57%  struct malloc_chunk
     0.54%  struct av_decision
     0.41%  struct btrfs_path
     0.36%  struct av_decision*
     0.34%  unsigned char
     0.32%  struct hlist_bl_head
```

type, symbol

```
# perf report --stdio -s type,sym -i perf.data.mem.find
# Total Lost Samples: 0
#
# Samples: 25K of event 'cpu_core/mem-loads-aux/'
# Event count (approx.): 170070020
#
# Overhead  Data Type              Symbol
# ........  .........              ..............................
    12.56%  struct btrfs_key       [k] btrfs_real_readdir
     7.40%  int                    [.] __GI___readdir64
     5.98%  unsigned int           [k] _raw_spin_lock
     4.75%  (stack operation)      [k] locks_remove_posix
     3.24%  (stack operation)      [k] btrfs_verify_level_key
     2.77%  (stack operation)      [k] check_buffer_tree_ref
     2.76%  struct                 [k] up_read
     2.47%  struct extent_buffer*  [k] btrfs_search_slot
     2.46%  struct __va_list_tag   [.] __printf_buffer
     2.42%  struct btrfs_key       [k] btrfs_comp_cpu_keys
     2.07%  struct                 [k] down_read
     1.81%  struct extent_buffer   [k] release_extent_buffer
     1.59%  (unknown)              [k] memcpy
     1.56%  struct                 [k] check_buffer_tree_ref
     1.24%  (unknown)              [k] __srcu_read_unlock
     1.16%  struct inode           [k] generic_fillattr
     1.14%  unsigned int           [k] find_extent_buffer_nolock
     1.14%  (stack canary)         [k] locks_remove_posix
     1.04%  struct                 [k] __fput_sync
     1.01%  struct _ftsent         [.] fts_compare_ino.lto_priv.0
     0.97%  long long unsigned int [k] mutex_lock
     0.93%  struct _ftsent         [.] consider_visiting
     0.89%  (stack canary)         [k] fsnotify
     0.86%  (stack operation)      [k] read_extent_buffer
     0.83%  (unknown)              [k] __srcu_read_lock
     0.83%  (stack operation)      [k] __btrfs_tree_read_lock
     0.81%  long long unsigned int [k] lockref_put_return
     0.79%  (unknown)              [.] __memmove_avx_unaligned_erms
     0.76%  (stack canary)         [k] btrfs_verify_level_key
```

FIELDS

```
# perf report -s type,typeoff --hierarchy --stdio -i perf.data.mem.find
#
#    Overhead  Data Type / Data Type Offset
SNIP
     2.15%     struct inode
        0.26%     struct inode  +40 (i_sb)
        0.21%     struct inode +356 (i_readcount.counter)
        0.15%     struct inode  +56 (i_security)
        0.15%     struct inode  +13 (i_flags)
        0.12%     struct inode   +8 (i_gid.val)
        0.12%     struct inode +360 (i_fop)
        0.11%     struct inode   +4 (i_uid.val)
        0.10%     struct inode  +72 (i_nlink)
        0.09%     struct inode  +88 (__i_atime.tv_sec)
        0.09%     struct inode  +32 (i_op)
        0.09%     struct inode   +0 (i_mode)
        0.09%     struct inode  +64 (i_ino)
        0.08%     struct inode  +12 (i_flags)
        0.07%     struct inode +112 (__i_mtime.tv_nsec)
        0.07%     struct inode +144 (i_blocks)
        0.06%     struct inode  +96 (__i_atime.tv_nsec)
        0.05%     struct inode  +80 (i_size)
        0.05%     struct inode  +76 (i_rdev)
        0.05%     struct inode +128 (__i_ctime.tv_nsec)
        0.04%     struct inode +120 (__i_ctime.tv_sec)
        0.04%     struct inode +140 (i_bytes)
        0.04%     struct inode +104 (__i_mtime.tv_sec)
        0.03%     struct inode +142 (i_blkbits)
SNIP
```

HIERARCHY

```
# perf report -s type,typeoff,sym --hierarchy --stdio -i perf.data.mem.find
SNIP
    15.35%        struct btrfs_key
        7.05%        struct btrfs_key +0 (objectid)
           6.04%        [k] btrfs_real_readdir
           0.76%        [k] btrfs_comp_cpu_keys
           0.26%        [k] btrfs_bin_search
        4.27%        struct btrfs_key +9 (offset)
           3.31%        [k] btrfs_real_readdir
           0.94%        [k] btrfs_comp_cpu_keys
           0.02%        [k] btrfs_bin_search
        4.03%        struct btrfs_key +8 (type)
           3.21%        [k] btrfs_real_readdir
           0.73%        [k] btrfs_comp_cpu_keys
           0.09%        [k] btrfs_bin_search
SNIP
```

ANNOTATE EXAMPLE

```
# perf annotate --stdio --data-type
Annotate type: 'struct btrfs_key' in [kernel.kallsyms] (6282 sample
 event[0] = cpu_core/mem-loads-aux/
 event[1] = cpu_core/mem-loads,ldlat=30/P
============================================================
        Percent  offset  size  field
 100.00  100.00       0    17  struct btrfs_key         {
  45.93   45.90       0     8      __u64       objectid;
  26.26   26.52       8     1      __u8        type;
  27.80   27.58       9     8      __u64       offset;
                                 };
```

PACKED

```
root@number:~# strace -e openat pahole btrfs_key |& tail -11
openat(AT_FDCWD, "/sys/kernel/btf/vmlinux", O_RDONLY) = 3
struct btrfs_key {
        __u64                objectid;       /*  0  8 */
        __u8                 type;           /*  8  1 */
        __u64                offset;         /*  9  8 */

        /* size: 17, cachelines: 1, members: 3 */
        /* last cacheline: 17 bytes */
} __attribute__((__packed__));

+++ exited with 0 +++
root@number:~#
```

THE STEPS

```
# perf --debug type-profile annotate --data-type
find data type for 0x6(reg7) at intel_pmu_handle_irq+0x53
CU for arch/x86/events/intel/core.c (die:0x1b1f23)
frame base: cfa=1 fbreg=7
found "late_ack" in scope=1/1 (die: 0x1da6df) stack_offset=0x60 type
 variable location: use frame base, offset=0xfffffffffffffa6
 type='_Bool' size=0x1 (die:0x1b21d4)
```

THE STEPS

```
# perf --debug type-profile annotate --data-type
find data type for 0x6(reg7) at intel_pmu_handle_irq+0x53
CU for arch/x86/events/intel/core.c (die:0x1b1f23)
frame base: cfa=1 fbreg=7
found "late_ack" in scope=1/1 (die: 0x1da6df) stack_offset=0x60 type
 variable location: use frame base, offset=0xfffffffffffffa6
 type='_Bool' size=0x1 (die:0x1b21d4)
```

```
 static int intel_pmu_handle_irq(struct pt_regs *regs)
{
        struct cpu_hw_events *cpuc = this_cpu_ptr(&cpu_hw_events);
        bool late_ack = hybrid_bit(cpuc->pmu, late_ack);
        bool mid_ack = hybrid_bit(cpuc->pmu, mid_ack);
        int loops;
```

ANOTHER

```
find data type for 0(reg1, reg0) at arch_asym_cpu_priority+0x1b
CU for arch/x86/kernel/itmt.c (die:0xed3cc9)
frame base: cfa=1 fbreg=7
scope: [1/1] (die:ed5101)
bb: [0 - 1b]
var [0] reg5 type='int' size=0x4 (die:0xed3d3e)
mov [9] reg5 -> reg5 type='int' size=0x4 (die:0xed3d3e)
mov [c] imm=0x19a38 -> reg0
mov [13] percpu base reg1
chk [1b] reg1 offset=0 ok=0 kind=2 cfa
no variable found
```

ANOTHER

```
find data type for 0(reg1, reg0) at arch_asym_cpu_priority+0x1b
CU for arch/x86/kernel/itmt.c (die:0xed3cc9)
frame base: cfa=1 fbreg=7
scope: [1/1] (die:ed5101)
bb: [0 - 1b]
var [0] reg5 type='int' size=0x4 (die:0xed3d3e)
mov [9] reg5 -> reg5 type='int' size=0x4 (die:0xed3d3e)
mov [c] imm=0x19a38 -> reg0
mov [13] percpu base reg1
chk [1b] reg1 offset=0 ok=0 kind=2 cfa
no variable found
```

```
int arch_asym_cpu_priority(int cpu)
{
        return per_cpu(sched_core_priority, cpu);
}
```

bpf_map

```
$ perf annotate --data-type=bpf_map --stdio
Annotate type: 'struct bpf_map' in [kernel.kallsyms] (4 samples):
 event[1] = cpu_core/mem-loads,ldlat=30/P
============================================================
 Percent  offset  size  field
  100.00       0   256  struct bpf_map   {
   63.12       0     8      struct bpf_map_ops* ops;
    0.00       8     8      struct bpf_map*     inner_map_meta;
    0.00      16     8      void*       security;
    0.00      24     4      enum bpf_map_type   map_type;
   36.88      28     4      u32 key_size;
    0.00      32     4      u32 value_size;
    0.00      36     4      u32 max_entries;
    0.00      40     8      u64 map_extra;
    0.00      48     4      u32 map_flags;
    0.00      52     4      u32 id;
    0.00      56     8      struct btf_record*  record;
    0.00      64     4      int numa_node;
    0.00      68     4      u32 btf_key_type_id;
SNIP
```

USE BTF?

- If DWARF not available
- BTF Type info
- per-cpu variables in BTF
- kallsyms
- Kernel functions using registers as args
- DECL_TAGs for kfuncs: args

COMPANION BTF

- For kernel analysis needs
- A BTF -debuginfo package?
- Extra file in kernel package?
- bpf_line_info for vmlinux, modules
- Now just in BPF programs ELF files