Overflowing with Strings

A new(ish) dimension in DWARF size concerns

Intro

Initially presenting as corrupted names in backtraces

Eventually eyeballed it

— — —

```
$ llvm-objdump -h tensorflow_model_server.dwo
tensorflow_model_server.dwp:    file format elf64-x86-6
Sections:
Idx Name                    Size     VMA              Type
  0                         00000000 0000000000000000
  1 .debug_abbrev.dwo       037dd80f 0000000000000000
  2 .debug_line.dwo         01e96dfe 0000000000000000
  3 .debug_loclists.dwo     1f554e70 0000000000000000
  4 .debug_str_offsets.dwo  1e818bf8 0000000000000000
  5 .debug_rnglists.dwo     1a17987a 0000000000000000
  6 .debug_info.dwo         6ab014bd 0000000000000000
  7 .debug_str.dwo          1058a03cb 0000000000000000
  8 .debug_cu_index         0012faa8 0000000000000000
  9 .debug_tu_index         051e3a28 0000000000000000
 10 .shstrtab               000000ab 0000000000000000
```

Unmangled Names, Expression Templates

- Eigen (linear algebra library) uses expression templates
- Eg:

```
TensorEvaluator<const Eigen::TensorSlicingOp<
                    const Eigen::DSizes<int, 1>, const Eigen::DSizes<int, 1>,
                    const Eigen::TensorMap<
                        Eigen::Tensor<const std::complex<float>, 1, 1, int>, 16,
                        MakePointer> >,
                Eigen::ThreadPoolDevice>
```

- Largest name is over 50,000 characters

Source Solutions

Shorten Names

- 50710649 instances of ‘Eigen’ in names
- renamed ‘Eigen’ to ‘E’
- Save 50710649 * 4 (`igen`) bytes, 0xC172134 (~200MB)
- Would get below the limit

__attribute__((nodebug)) (credit to rnk@)

- Previously used this to remove type traits typedefs from libc++
- Doesn’t currently apply to class types - could make it do that & omit variables of these types from DWARF
- Would mean these would be harder to debug (unclear if that’s a significant loss/issue)

DWARF Solutions

DWARF64

- Recently implemented in LLVM
- Maybe relatively untested (in LLVM, and in general)
- Makes everything bigger

Simple template names

```
DW_TAG_structure_type
  DW_AT_name      ("t1<float>")
  …
  DW_TAG_template_type_parameter
    DW_AT_type    (0x000000ff "float")
    DW_AT_name    ("T")
  NULL
```

```
DW_TAG_structure_type
  DW_AT_name      ("t1")
  …
  DW_TAG_template_type_parameter
    DW_AT_type    (0x000000ff "float")
    DW_AT_name    ("T")
  NULL
```

Simple template names

- Simple type name, structural template parameters
- Seems to work with GCC
- Doesn’t work with LLDB
- Reduced a particularly problematic binary’s dwp
  - .debug_str.dwo by 42%
  - Overall .dwp size by 25%
- Only 25% str/10% overall reduction for another, less expression-template-heavy but otherwise large program

Omit or share mangled names

```
DW_TAG_subprogram
  DW_AT_linkage_name ("_Z2f1IiEvv")
  DW_AT_name         ("f1<int>")
  ...

  DW_TAG_template_type_parameter
    DW_AT_type       (0x00000066 "int")
    DW_AT_name       ("T")
```

```
DW_TAG_subprogram
  DW_AT_linkage_name ("_Z2f1IiEvv")
  DW_AT_name         ("f1<int>")
  ...

  DW_TAG_template_type_parameter
    DW_AT_type       (0x00000066 "int")
    DW_AT_name       ("T")
```

```
DW_TAG_subprogram
  DW_AT_linkage_name (0x6a8d7f8fda3f5245)
  DW_AT_name         ("f1<int>")
  ...

  DW_TAG_template_type_parameter
    DW_AT_type       (0x00000066 "int")
    DW_AT_name       ("T")
```

Omit or share mangled names

- Omit mangled name, rebuild from structural representation
  OR
- Keep mangled names in a separate, compressed section, refer to them by hash (share approach with ELF symbol table)
- Would need to teach all consumers new tricks
- Reduced a particularly problematic binary's dwp
  - .debug_str.dwo by 52%
  - Overall .dwp size by 33%
- 56% str/25% overall reduction for another, less expression-template-heavy but otherwise large program

Progress

Simple template names

- Implementing in Clang/LLVM
- Added -gsimple-template-names={none,simple,mangled}
  - None: status quo
  - Simple: Proposed solution
  - Mangled: _STN<full name> - flag a name as “should be able to be simplified, but include the full name anyway”
- Added functionality to llvm-dwarfdump --verify
  - If a Mangled name is detected, rebuild the full name from the structural representation and compare it to the full name provided
- Roundtrips clang, llvm-dwarfdump, and some internal binaries without errors

Simple template names

- Skipped some names for simplicity:
  - Operator overload names (ambiguity “operator t1<int>” - is this a templated conversion to t1, or a conversion to t1<int>? How to know whether it’s a simplified name that needs template parameters added, or is it the full name?)
  - Atomics (DWARF has no representation of _Atomic(int) V int, but they do produce distinct template instantiations)
  - Anonymous enums/structs/lambdas (contain source locations in the name, lambdas are ambiguous)

Simple template names

- Skipped some names for simplicity:
  - Vector types (possible, but extra DWARF to look at)
  - Pointer non-type-template parameters
  - Integral non-type-template parameters > 64 bits

Simple template names

- Found a bunch of bugs/inconsistencies in clang's type printing:
  - Missing/extra whitespace around template parameter packs
  - Type suffixes on integer non-type-template parameters (intentionally only used when neeeded, but makes them hard to reconstitute exactly)
  - Use of "preferred name" of templates
  - Qualification of partially qualified template template parameters