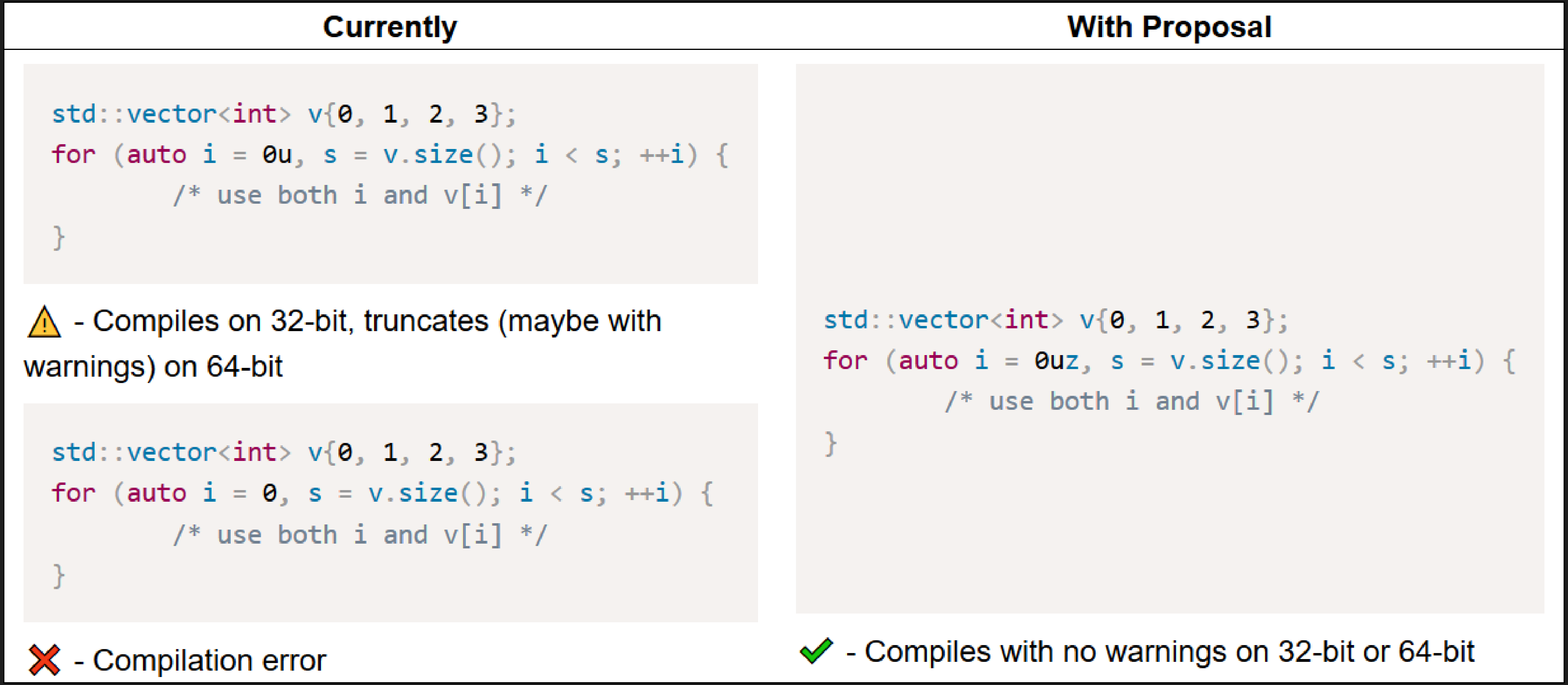
Literal suffixes for `size_t` and `ptrdiff_t`

P0330 – A C++20 Paper

JeanHeyd Meneide – phdofthehouse@gmail.com – Twitter: @thephantomderp

https://thephd.github.io/vendor/future_cxx/papers/d0330.html

Before/After



Currently

```
std::vector<int> v{0, 1, 2, 3};
for (auto i = 0u, s = v.size(); i < s; ++i) {
        /* use both i and v[i] */
}
```

⚠ - Compiles on 32-bit, truncates (maybe with warnings) on 64-bit

```
std::vector<int> v{0, 1, 2, 3};
for (auto i = 0, s = v.size(); i < s; ++i) {
        /* use both i and v[i] */
}
```

✗ - Compilation error

With Proposal

```
std::vector<int> v{0, 1, 2, 3};
for (auto i = 0uz, s = v.size(); i < s; ++i) {
        /* use both i and v[i] */
}
```

✓ - Compiles with no warnings on 32-bit or 64-bit

History

- Originally proposed as Library User-Defined Literals in C++17 timeframe
  - Written by Rein Halbersma, presented by Walter Brown
- Last-minute course-change from LWG, just before it was actually accepted:
  - *"This is not a library concern (any more); EWG has taken ownership. The types in question (size_t and ptrdiff_t) are not library types, but rather core language types."* – LWG, 2018

Design

- Uses the suffix z
  - Cannot use s: is in clash with upcoming sf – short float paper
  - Avoid sz: clash with short float that makes sz harder to understand

- zu/uz – size_t
  - Core language allows u to appear on either side of a literal
  - Equivalent to type of `decltype(sizeof(0))`

- z – ptrdiff_t
  - Equivalent to type of `decltype((char*)nullptr - (char*)nullptr)`

What about t and z?

- Some advocated for t (ptrdiff_t) and uz/zu (size_t) to be separate suffixes
  - If used by themselves, then we leave a problem with ut/tu and plain z
  - unsigned ptrdiff_t and signed size_t do not exist in the standard (!!)

-
```
int main() {
     signed decltype(sizeof(0)) x = 0;
     unsigned decltype((char*)nullptr - (char*)nullptr) y = 0;
     return x - y;
}
```

The Type You Are Looking For is in Another ~~Castle~~Standard

- main.cpp:2:32: warning: long, short, signed or unsigned used invalidly for 'x' [-Wpedantic]
  main.cpp:3:56: warning: long, short, signed or unsigned used invalidly for 'y' [-Wpedantic]

- C++ does not define what these types are
  - That is a much more complicated and involved paper
  - Not even POSIX defines what an `unsigned ptrdiff_t` would be (you do have `ssize_t`)

Wording Complete

- Paper is Core-ready
  - Reviewed by Tim Song prior to meeting

Polls

- Forward to EWG and CWG to handle for C++20?

Strongly in Favor	In Favor	Neutral	Against	Strongly Against

Extra Polls (if needed)

- Have `t` for `ptrdiff_t` and `uz/zu` for size_t, then leave it up to somebody to figure out what `ut/tu` for `unsigned ptrdiff_t` and `z` for `signed size_t?`

Strongly in Favor	In Favor	Neutral	Against	Strongly Against