ABSTRACT
Scaling OpenMP with LLVM for Exascale performance and portability (SOLLVE) aims to scale OpenMP by leveraging LLVM for exascale performance and portability of applications. A Verification & Validation (V&V) testsuite tests various OpenMP directives to evaluate system & vendor compliance. The V&V suite is run on various systems, including NERSC’s Perlmutter system & Oak Ridge National Laboratory’s Summit & Crusher systems.

BACKGROUND INFO
OpenMP, a parallel-based programming model, allows for performance optimization in C, C++, and Fortran, with its features called “directives” listed in the OpenMP specification (spec). The purposes of the SOLLVE Verification and Validation (V&V) testsuite are to:
• Evaluate compiler’s compliance with the specification
• Identify ambiguities in the specification
• Illustrate a system’s ability to run OpenMP directives & utilize offloading parallel directives on GPUs
• Demonstrate the use & purpose of new OpenMP directives to application developers

OpenMP is useful for many application developers working on HPC systems to ensure their code is running at maximum efficiency. Our testsuite ensures that the OpenMP specification, compiler vendors & system operators are implementing OpenMP effectively.

APPROACH
• Tests are usually written to ensure the test would fail if the directive were to not work properly.
• If the test fails, analysis is done to determine the issue:
  • The test was written improperly
  • The directive has not yet been implemented
  • The specification is too ambiguous
• If the test passes, it is merged into the repository.
• Results for each system & compiler are then uploaded to the website.

TEST EXAMPLE
```c
#pragma omp parallel num_threads(threads)
while(1){
  int tot;
  #pragma omp atomic read
  tot = total;  
  if (tot <= 0)  
    break;
  #pragma omp masked
  {
    OMPVV_TEST_AND_SET_VERBOSE(errors, omp_get_thread_num() != 0); // primary thread
    ct++;
    #pragma omp atomic
    total = total-1;  
  }
  OMPVV_TEST_AND_SET_VERBOSE(errors, ct != 10);  
}
```

Figure 1: test_masked.c code segment
This test, which focuses on the omp masked directive which only runs on the master thread, demonstrates:
- OMPVV_TEST_AND_SET_VERBOSE function, which is part of the V&V suite and is a primary tool in reporting errors for test results
- Errors reported if the running thread is not the master (first) thread
- Errors reported if masked code segment does not run 10 times

RESULTS ON PERLMUTTER, SUMMIT & CRUSHER
Figure 2 shows GCC & Clang perform best on Perlmutter, while NVC performs the worst, with more failing tests than passing. This is interesting, as the NVHPC compiler performs much better on other systems, such as Summit (Figure 3). It is important to note that LLVM’s Clang does not include a fortran compiler.

Figure 3 shows GCC performing the best on the Summit system. Compared to Perlmutter, GCC has over 100 more passing tests on the system, despite it using a slightly downgraded version of the compiler. In contrast to perlmutter, LLVM’s clang compiler performs the worst on Summit.

CONCLUSIONS
- In total, we have so far written 289 C, 26 C++ and, 172 Fortran tests
- The V&V suite targets up to OpenMP 5.2 specification
- The suite runs on pre-exascale systems (spock, crusher), Perlmutter and Argonne JLSE test beds

FUTURE WORK
- Working on new tests for the latest OpenMP specification (5.2)
- Expand our Fortran tests so that more C/C++ tests have Fortran counterparts
- Running/Testing the suite on more machines to further test the specification

CALLS TO ACTION
GitHub V&V repository (https://github.com/SOLLVE/solve-vv) is open for download, logging issues, suggesting tests, etc.

ACKNOWLEDGEMENTS
Thanks to OLCF: This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.
Thanks to ECP: This research was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.
Thanks to NERSC: This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231.