Item 33
git_comments:

1. 8 Test - special case, begin inside last slice block but outside asset len

git_commits:

1. **summary:** slice/handleFirstServerHeader: return sooner on requested range errors (#7486)
   message: slice/handleFirstServerHeader: return sooner on requested range errors (#7486) (cherry picked from commit df01ace2b0bdffd3ddcc5b2c7587b6d6fed5234c)

github_issues:

github_issues_comments:

github_pulls:

1. **title:** slice/server: handleFirstServerHeader exit sooner on detected requested range errors.
   body: When examining the first server response header with a bad range request the function should exit sooner. I was able to recreate a crash and added a test case for it.

github_pulls_comments:

1. Why is that? I've never heard that claim before. Testing with g++ back to 8.3, `std::numeric_limits<in64_t>::max()` works as a `constexpr`, e.g. use of it causes a load immediate instruction, even at "-O0". If you want to be absolute certain, do this: ``` #include <type_traits> static_assert(std::integral_constant<int64_t, Range::maxval>::value); ``` If that compiles, it's a compile time constant.
2. I believe per the standard numeric_limits are constexpr. https://en.cppreference.com/w/cpp/types/numeric_limits/max lists it as constexpr. I'd say numeric_limits are more idiomatic for C++ than the _MAX macros. Out of curiosity, @traeak , did you run into an issue with numeric_limits that motivated this? Thanks,
3. Looks like the handleFirstServerHeader should return sooner when range errors are detected. backtrace seemed to indicate that this static var pointed to the same address as the first data member in that Range instance. That sent me in the wrong direction. std::numeric_limits has sometimes not behaved not very well with msvc, especially when used for default arg values.
4. Cherry-picked to v9.0.x branch. Cherry-picked to v9.1.x branch.

github_pulls_reviews:

1. These probably get optimized out but it's worthwhile cleanup.
2. It's not clear why this is being changed from max / 4 to max / 2, or really why it's not just max. But I'll take it on faith.
3. it just has to be sufficiently large but not max64.

jira_issues:

jira_issues_comments: