Debug Output Performance

for TS execution tracing

Problem: When debug output is globally enabled, even if no debug tag is enabled, TS performance is degraded. This is because most Debug output calls (when globally enabled) must test if there tag matches the configured regular expression for enabled tags.

Summary of Solution: Maintain an std::set whose elements contain all unique debug tags with an "enabled" flag for each. When the global enable is true, each debug call will have a pointer to the element in the std::set for its tag, so it can simply check the "enabled" flag. Whenever the regular expression for enabled tags is changed, all entries in the std::set are iterated over, and each entry's "enabled" flag is set based on whether the entry's tag matches the new regular expression. (PR #7674)

Plugins

- New TS API function TSDbgCtlCreate(char const *tag) returns const pointer to TSDbgCtl. (It's a pointer into the std::set mentioned above.)
- New TS API function TSDbg() deprecates TSDebug(). TSDbg() takes TSDbgCtl as first parameter instead of tag string.
- Setting proxy.config.diags.debug.enabled to 1 enables output from both TSDebug() and TSDbg() calls corresponding to tags matching the regex given by proxy.config.diags.debug.tags.
- TSDebug() calls still do a regex match on each call (impacting performance). Therefore, a new value of 3 for debug.enabled is supported. It only enables output from TSDbg() calls, not from TSDebug() calls.

Core TS

- New definitions/declarations added to tscore/Diags.h.
- New class DbgCtl (whose instances should be created statically). Instances constructed with a debug tag, contains a pointer to the entry in the std::set for the tag.
- New macro Dbg(), like Debug() except it takes a DbgCtl instance as its first parameter instead of a tag string.
- Debug() macro changed so it defines a static instance of DbgCtl at each location where it is invoked. Thus it also avoids doing a regex match when debug output is enabled. Using Debug() instead of Dbg() slightly increases TS memory use, and slightly increases CPU cache evictions.
- Dbg() and Debug() are both enabled by a setting of either 1 or 3 for debug.enabled.
- Analogous changes for the dumpster fire of alternative ways of generating debug output in core.
- The PR currently includes many changes of Debug() calls into Dbg() calls. These changes are not strictly necessary.
- Since (with this change) checking if a tag is enabled only involves two memory fetches and a few instructions, the global enable for core seems potentially superfluous. But the global enable check has the advantage of using a single memory location that is very L1 cache-hot. Performance testing results were ambiguous with regard to this.

TS Throughput Testing == Hell

- Compared to an RTOS (VxWorks) I use to work on, repeating the same performance test on Linux can give very inconsistent results. But this may be due in part to difference between embedded processors and server/desktop processors.
- When testing with h2load, PUSHing an object to serve into cache seems to lead to more consistent results than using generator.so.
- Repeatability of throughput testing in production is impared by traffic variations, and perhaps by load balancer behavior.