A Study of Dynamic Software Update Quiescence for Multithreaded Programs
Christopher M. Hayden, Karla Saur, Michael Hicks, Jeffrey S. Foster
University of Maryland

Software Upgrades: Useful

Dynamic Software Updating (DSU)
Goal: Update programs while they run

Multithreaded Dynamic Software Updating
We support updates at developer-identified quiescent points, which are program locations that are reached between iterations of event-processing loops and at which point there is typically less in-flight state.

 void +thread_entry(void +arg) {
   // thread init code +
   +while (1) {
     +  qbensch_update(); // update point +
     +  loop body: typically handles a single program event +
   +}
}

The state in which all program threads have reached an update point is full quiescence, allowing an update to take place.

Goal: Reach full quiescence as quickly as possible.

Challenges:
Achieving full quiescence may be delayed or thwarted by blocking calls (waiting for I/O, waiting for condition variables).

Example: One thread could hold a mutex when it reaches its update point, but then another thread could block on the same mutex prior to reaching its own update point, delaying full quiescence indefinitely.

Solution: QBench Library
QBench provides calls that allow blocking calls to be interrupted.

Blocking I/O Calls: QBench’s main signal handler sends a signal to any other thread that has not already reached its update point and is not waiting on a condition variable.

Blocking Condition Variables: QBench replaces the pthread conditional wait call with a call that notes the condition variable argument in the global list of threads so that it can later be signaled by another thread once an update has been requested.

QBench Library Calls to Expediate Quiescence

void +thread_entry(void +arg) {
  // thread init code +
  +while (1) {
    +  qbensch_update(); // update point +
    +  pthread_mutex_unlock(&mutex);
    +  if (qbensch_update_requested) {
      +    continue; // reaches qbensch_update +
      +    ... handle connection +
    +  }
  +}
}

Results: Few Changes Necessary
We tested programs covering a wide range of domains including media streaming, caching, intrusion detection, and gaming.

The changes required to support full quiescence were small:

Results: Time to Quiescence < 1ms on average
In some cases quiescence would not have occurred without changes. The QBench library improved times to reach quiescence in all cases.

Dynamic Software Updating (DSU)
Goal: Update programs while they run

At some point during execution of a program, update to the new version. DSU must preserve and update existing program state such as existing connections, important data on the stack and heap, and the program counter.