Multithreaded Servers

- Servers must take advantage of concurrency to handle their client loads
- Usual approach is multithreading
- Performance at the cost of correctness
- Thread accesses to every shared resource must be properly synchronized
- Locking is dangerous too—deadlock, livelock, priority inversion, starving, etc.

Event-driven Servers

- Program registers interest in events (callbacks)
- Event loop waits for events; invokes handlers
- State stored in "context" which is passed as an argument when a handler is invoked
- No synchronization required
- Handlers are atomic blocks
- Single threaded
- Must use asynchronous calls; blocking stops progress
- Difficult to exploit multiprocessors

Approach

- Add concurrency without requiring synchronization
- Run event handlers in parallel when safe to do so
- Conservatively determine whether handlers share data unsafely
- Generate constraints on concurrent execution of handlers
- Provide detailed feedback—why do handlers conflict?

Runtime System

- Run handlers concurrently subject to the constraints generated by the analysis

Future Work

- Program analyzer and runtime system in active development (using CIL and libevent)
- Evaluation (on thttpd)
- Beyond event-driven programs—multithreading