Other languages: 繁體中文

Benchmarks

As a C++ Http application framework, performance should be one of the focus of attention. This section introduces Drogon's simple tests and achievements;

Test environment

- The system is Linux CentOS 7.4;
- The device is a Dell server, the CPU is two Intel(R) Xeon(R) CPUs E5-2670 @ 2.60GHz, 16 cores and 32 threads;
- Memory 64GB;
- gcc version 7.3.0;

Test plan and results

We just want to test the performance of the drogon framework, so we want to simplify the controller's processing as much as possible. We only do an HttpSimpleController and register it on the `/benchmark` path. The controller returns `<p>Hello, world!</p>` for any request. Set the number of drogon threads to 16. The processing function is as follows and you can find the source code at the `drogon/examples/benchmark` path:

```
void BenchmarkCtrl::asyncHandleHttpRequest(const HttpRequestPtr &req,
std::function<void (const HttpResponsePtr &)> &&callback)
{
    //write your application logic here
    auto resp = HttpResponse::newHttpResponse();
    resp->setBody("<p>Hello, world!</p>");
    resp->setExpiredTime(0);
    callback(resp);
}
```

For comparison, I chose nginx for comparison testing, wrote a `hello_world_module`, and compiled it with the nginx source. The nginx worker_processes parameter is set to 16.

The test tool is `httpress`, a good performance HTTP stress test tool.

We adjust the parameters of httpress, test each set of parameters five times, and record the maximum and minimum values of the number of requests processed per second. The test results are as follows:

Command line	Description	Drogon(kQPS)	nginx(kQPS)
httpress -c 100 -n 1000000 -t 16 -k -q URL	100 connections, 1 million requests, 16 threads,Keep-Alive	561/552	330/329
httpress -c 100 -n 1000000 -t 12 -q URL	100 connections, 1 million requests, 12 threads, no Keep-Alive	140/135	31/49

Command line	Description	Drogon(kQPS)	nginx(kQPS)
httpress -c 1000 -n 1000000 -t 16 -k -q URL	1000 connections, 1 million requests, 16 threads,Keep-Alive	573/565	333/327
httpress -c 1000 -n 1000000 -t 16 -q URL	1000 connections, 1 million requests, 16 threads,no Keep-Alive	155/143	52/50
httpress -c 10000 -n 4000000 -t 16 -k -q URL	10000 connections, 4 million requests, 16 threads,Keep-Alive	512/508	316/314
httpress -c 10000 -n 1000000 -t 16 -q URL	10000 connections, 1 million requests, 16 threads,no Keep-Alive	143/141	43/40

As you can see, using the Keep-Alive option on the client side, drogon can process more than 500,000 requests per second in the case where a connection can send multiple requests. This score is quite good. In the case that each request initiates a connection, CPU time will be spent on TCP connection establishment and disconnection, and the throughput will drop to 140,000 requests per second, which is reasonable.

It's easy to see that drogon has a clear advantage over nginx in the above test. If someone does a more accurate test, please correct me.

The image below is a screenshot of a test:


Test Result

Next: Causal profiling with coz