C++ http框架调研

测试环境：服务端与客户端之间带宽100Mbps左右

httplib 框架

只需要添加头文件`httplib.h`到项目中即可使用。

使用简单，但性能相对较差。

```
#include "httplib.h"
```

连续请求个数	单个请求数据量	平均传输时间	吞吐量
10	30MB	2107ms	14.24MB/s
100	30MB	2063ms	14.54MB/s

多线程(100)并发请求

单个线程连续请求个数	单个请求数据量	平均传输时间	吞吐量
100	30KB	2.12ms	13.82MB/s
1000	30KB	2.19ms	13.37MB/s

多线程(100)并发请求，服务端固定32线程

单个线程连续请求个数	单个请求数据量	平均传输时间	吞吐量
100	30KB	2.17ms	13.52MB/s
1000	30KB	2.01ms	14.56MB/s

Drogon 框架

性能优异

需要先编译源码

```
git clone git@github.com:drogonframework/drogon.git
cd drogon
git submodule update --init
mkdir build
cd build
cmake ..
make && sudo make install
```

包含头文件

```
#include <drogon/drogon.h>
```

连续请求个数	单个请求数据量	平均传输时间	吞吐量
10	30MB	2095ms	14.32MB/s
100	30MB	2094ms	14.32MB/s

多线程(100)并发请求

连续请求个数	单个请求数据量	平均传输时间	吞吐量
100	30KB	2.04ms	14.35MB/s
1000	30KB	2.12ms	13.83MB/s

Beast框架

使用简单，安装Boost库即可

连续请求个数	单个请求数据量	平均传输时间	吞吐量
10	30MB	1853ms	16.19MB/s
100	30MB	2070ms	14.49MB/s

多线程(100)并发请求

连续请求个数	单个请求数据量	平均传输时间	吞吐量
100	30KB	2.18ms	13.57MB/s
1000	30KB	2.19ms	13.38MB/s

Seastar 框架

需要先编译源码

```
sudo ./install-dependencies.sh
./configure.py --mode=release --prefix=/usr/local
sudo ninja -C build/release install
```

使用较为复杂，尚未使用成功