WebTorrent tracker performance comparison

This is a performance comparison of several WebTorrent tracker implementations, using *aquatic_ws_load_test*. It is somewhat unrealistic, as it simulates a relatively small number of clients, each sending a large number of requests. Nonetheless, I think that it gives a useful indication of relative performance.

Setup

Tested tracker implementations

Tracker	URL	Commit
aquatic_ws	https://github.com/greatest-ape/aquatic	cf6a11f
wt-tracker	https://github.com/Novage/wt-tracker	400a436
openwebtorrent-tracker	https://github.com/OpenWebTorrent/openwebtorrent-tracker	67caa5d

The reference implementation (https://github.com/webtorrent/bittorrent-tracker) had to be excluded, since it doesn't natively support TLS and *aquatic_ws_load_test* requires it. Previous benchmarks indicate that it achieves less that half the throughput of wt-tracker.

Hardware

Hetzner CCX62: 48 dedicated vCPUs (AMD Milan Epyc 7003)

General software information

Software	Version
Ubuntu	20.04
Linux	5.15.0
rustc	1.59.0
GCC	9.4.0
nodejs	10.19.0

Additional commands: *ulimit -n 97816*

See appendix for configuration files used for load tester and tracker implementations.

Measurements

wt-tracker

Load test workers	Responses per second
16	31k
1	30k
2	29k

No matter the number of load test workers, tracker CPU utilisation was 100%. With 16 such workers, there were lots of connection errors (“Cannot assign requested address”).

openwebtorrent-tracker

Load test workers	Responses per second
16	9k
1	9k

No matter the number of load test workers, tracker CPU utilisation was 100%. There were lots of connection reset errors.

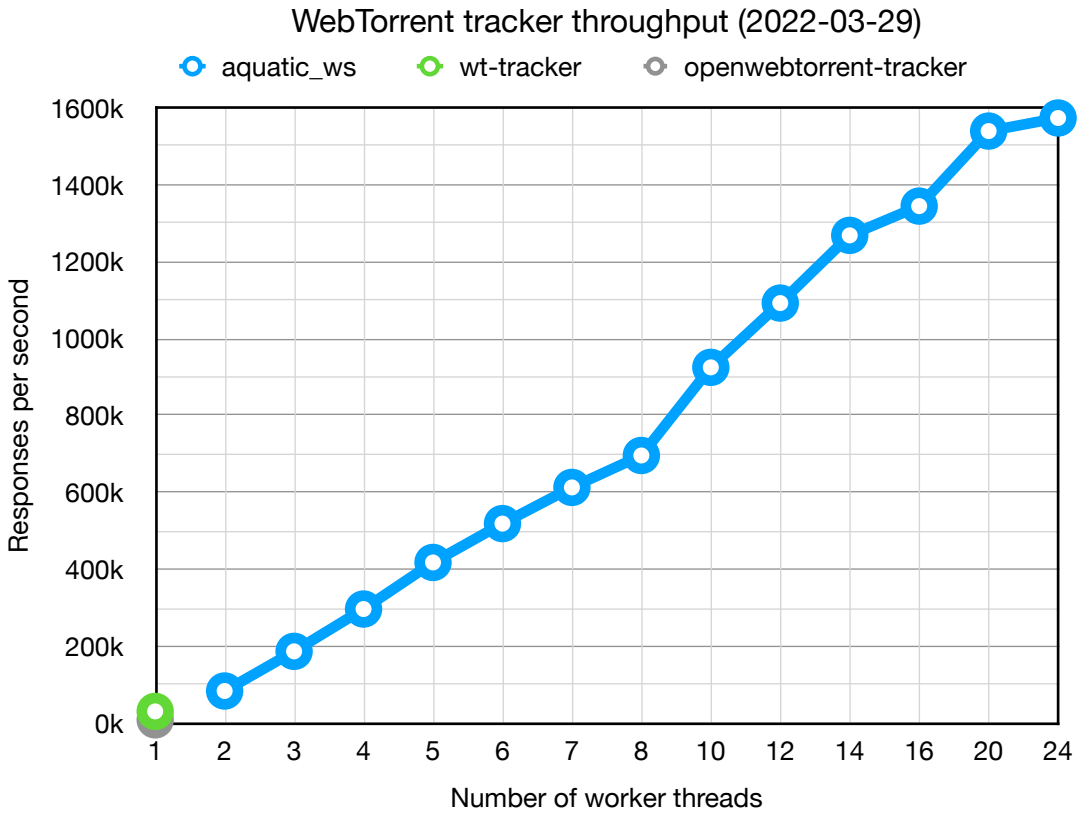
aquatic_ws

Results are ordered ascendingly by 1) total number of tracker workers and 2) number of responses per second. Best results within a worker number tier are marked in bold.

Tracker workers in total	Tracker socket workers	Tracker request workers	Responses per second
2	**1**	**1**	**84k**
3	**2**	**1**	**187k**
4	2	2	174k
4	**3**	**1**	**297k**
5	**4**	**1**	**418k**
6	4	2	392k
6	**5**	**1**	**519k**
7	5	2	501k
7	**6**	**1**	**613k**

Tracker workers in total	Tracker socket workers	Tracker request workers	Responses per second
8	6	2	624k
8	**7**	**1**	**696k**
10	8	2	875k
10	**9**	**1**	**925k**
12	11	1	777k
12	**10**	**2**	**1092k**
14	11	3	1047k
14	**12**	**2**	**1268k**
16	13	3	1277k
16	**14**	**2**	**1344k**
20	18	2	1265k
20	16	4	1303k
20	**17**	**3**	**1539k**
24	20	4	1530k
24	**21**	**3**	**1573k**

Results



aquatic_ws throughput (2022-03-29)

Number of worker threads	Responses per second
2	84k
3	187k
4	297k
5	418k
6	519k
7	613k
8	696k
10	925k
12	1092k
14	1268k
16	1344k
20	1539k
24	1573k

wt-tracker & openwebtorrent-tracker throughput (2022-03-29)

Tracker	Responses per second
wt-tracker	31k
openwebtorrent-tracker	9k

Appendix: application configuration

aquatic_ws_load_test

```
# aquatic_ws_load_test configuration

server_address = "127.0.0.1:3000"
log_level = "error"
num_workers = 16
num_connections_per_worker = 512
connection_creation_interval_ms = 10
duration = 60

[torrents]
offers_per_request = 2
number_of_torrents = 10000
torrent_selection_pareto_shape = 2.0
peer_seeder_probability = 0.25
weight_announce = 99
weight_scrape = 1
```

wt-tracker

```
{
  "servers": [{
      "server": {
        "port": 3000,
        "host": "127.0.0.1",
        "key_file_name": "key.pem",
        "cert_file_name": "cert.crt",
        "passphrase": "",
        "ssl_prefer_low_memory_usage": false
      },
      "websockets": {
        "path": "/*",
        "maxPayloadLength": 65536,
        "idleTimeout": 240,
        "compression": 1
      }
    }
  ],

  "tracker": {
    "maxOffers": 20,
    "announceInterval": 120
  },

  "websocketsAccess": {
  }
}
```

openwebtorrent-tracker

Application was invoked with:

./openwebtorrent-tracker --port 3000 --ssl-cert "cert.crt" --ssl-key "key.pem"

aquatic_ws

Default settings were used, except that TLS was set up and cleaning intervals were set to 600 seconds.