压测报告

一、软件组成

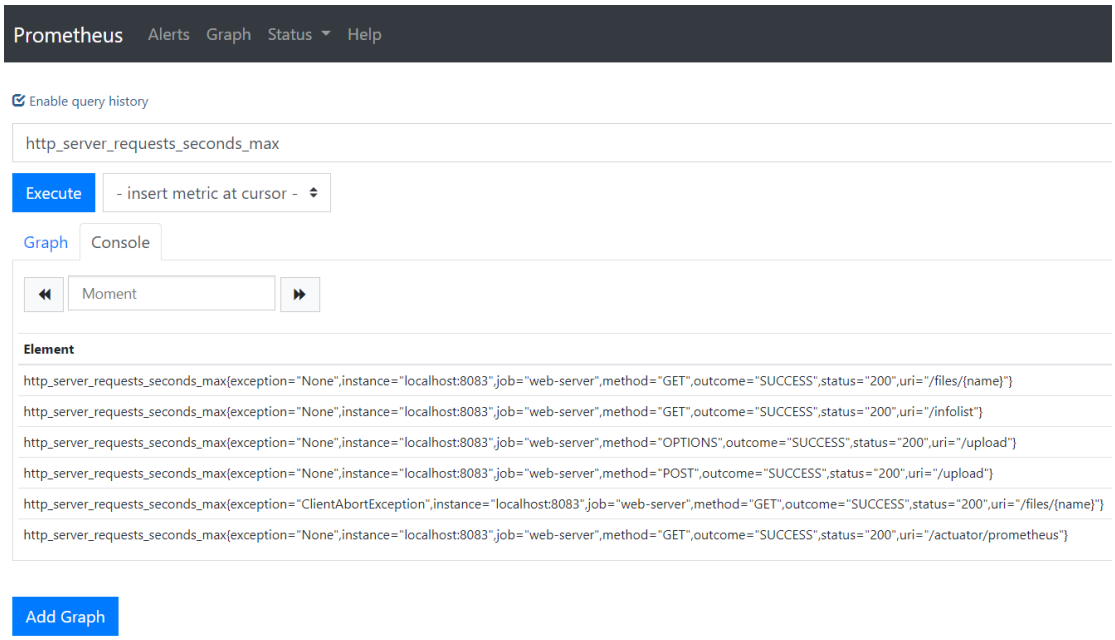
1. Gatling——负责发送 http 请求，输出服务器的响应情况。
   GatlingTestSimulation.scala:
   同时发送 400 个 http://localhost:8083/infolist 请求

```scala
class GatlingTestSimulation extends Simulation {

  val httpProtocol = http
    .baseUrl("http://localhost:8083")

  val infoes = exec(http("informations").get("/infolist"))
  // val files = exec(http("files").get("/files/360p_0 Grace_Trim.mp4"))

  val scn = scenario("TEST WEB-SERVER").exec(infoes)
  // val scn = scenario("TEST WEB-SERVER").exec(files)

  setUp(scn.inject(atOnceUsers(400)).protocols(httpProtocol))
  // setUp(scn.inject(rampUsers(600) over (3 seconds)).protocols(httpProtocol))

}
```

2. prometheus——负责性能监控，并根据一定的规则发出警报。
   prometheus.yml:
   alerting 的配置表示 prometheus 与 alertmanager 的通信地址为 localhost:9093，rule_files 的配置表示 prometheus 报警规则在 prometheus.rules.yml 中说明，scrape_configs 的配置表示 prometheus 的监控界面所处地址为 localhost:9090，它监控的目标是 web-server,该目标的性能数据所处的地址是 localhost:8083/actuator/prometheus

```yaml
# Alertmanager configuration
alerting:
  alertmanagers:
  - static_configs:
    - targets:
      - localhost:9093

# Load rules once and periodically evaluate them according to the global
rule_files:
  - "prometheus.rules.yml"
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries
  - job_name: 'prometheus'
```

```yaml
# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped
  - job_name: 'prometheus'

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
    - targets: ['localhost:9090']

  - job_name: web-server
    metrics_path: /actuator/prometheus
    static_configs:
    - targets: ['localhost:8083']
```



prometheus.rules.yml：
如果 http 请求的最大响应时间大于 0.1 秒，就发送一个警报给 alertmanager。

```yaml
groups:
- name: example
  rules:
  - alert: LongLatencyTime
    expr: http_server_requests_seconds_max{exception="None",method="GET",status="200"} > 0.1
    for: 5s
    labels:
      severity: page
    annotations:
      summary: long latency time, the system need to be extended
```

3. alertmanager——负责获取 prometheus 发出的警报，然后将警报发到个人邮箱。

```yaml
global:
  #超时时间
  resolve_timeout: 5m
  #smtp地址需要加端口
  smtp_smarthost: 'smtp.qq.com:465'
  smtp_from: '443160587@qq.com'
  #发件人邮箱账号
  smtp_auth_username: '443160587@qq.com'
  #账号对应的授权码（不是密码）
  smtp_auth_password: 'fthvkejwmguubieh'
  smtp_require_tls: false

route:
  group_by: ['alertname']
  group_wait: 10s
  group_interval: 1m
  repeat_interval: 4h
  receiver: 'mail'
receivers:
- name: 'mail'
  email_configs:
  - to: '443160587@qq.com'
```

---

1 alert for alertname=LongLatencyTime

View In AlertManager

[1] Firing

Labels

alertname = LongLatencyTime
exception = None
instance = localhost:8083
job = web-server
method = GET
outcome = SUCCESS
severity = page
status = 200
uri = /infolist

Annotations

summary = long latency time, the system need to be extended

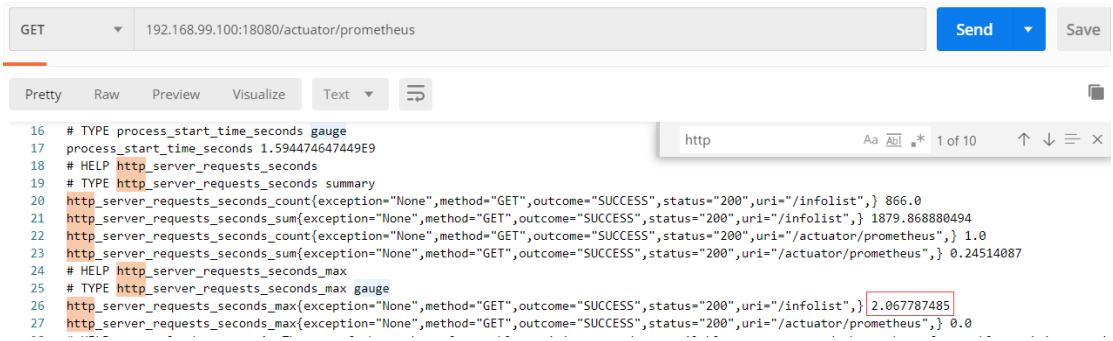
Source

二、实验过程及结果

2.1 1个服务器

```
Simulation gatlingtest.GatlingTestSimulation completed in 4 seconds
Parsing log file(s)...
Parsing log file(s) done
Generating reports...

================================================================================
---- Global Information --------------------------------------------------------
> request count                                        400 (OK=400    KO=0     )
> min response time                                   1948 (OK=1948   KO=-     )
> max response time                                   4529 (OK=4529   KO=-     )
> mean response time                                  3373 (OK=3373   KO=-     )
> std deviation                                        909 (OK=909    KO=-     )
> response time 50th percentile                       3347 (OK=3347   KO=-     )
> response time 75th percentile                       4303 (OK=4303   KO=-     )
> response time 95th percentile                       4475 (OK=4475   KO=-     )
> response time 99th percentile                       4520 (OK=4520   KO=-     )
> mean requests/sec                                     80 (OK=80     KO=-     )
---- Response Time Distribution ------------------------------------------------
> t < 800 ms                                            0 (  0%)
> 800 ms < t < 1200 ms                                  0 (  0%)
> t > 1200 ms                                         400 (100%)
> failed                                                0 (  0%)
================================================================================
```

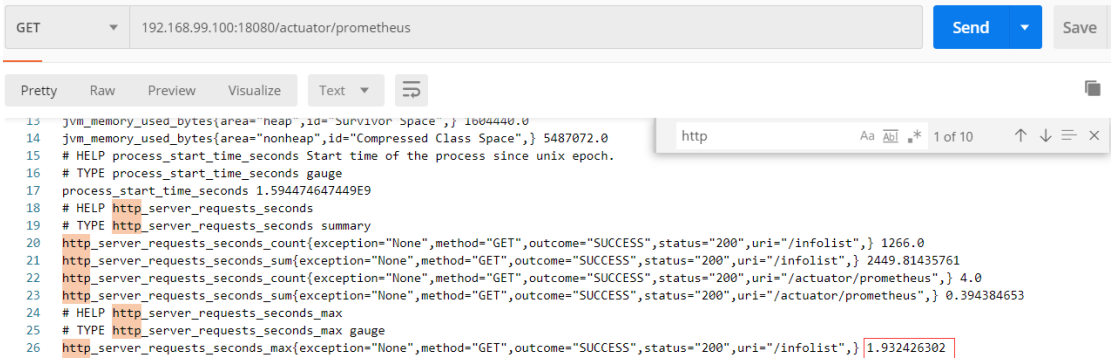
server0:



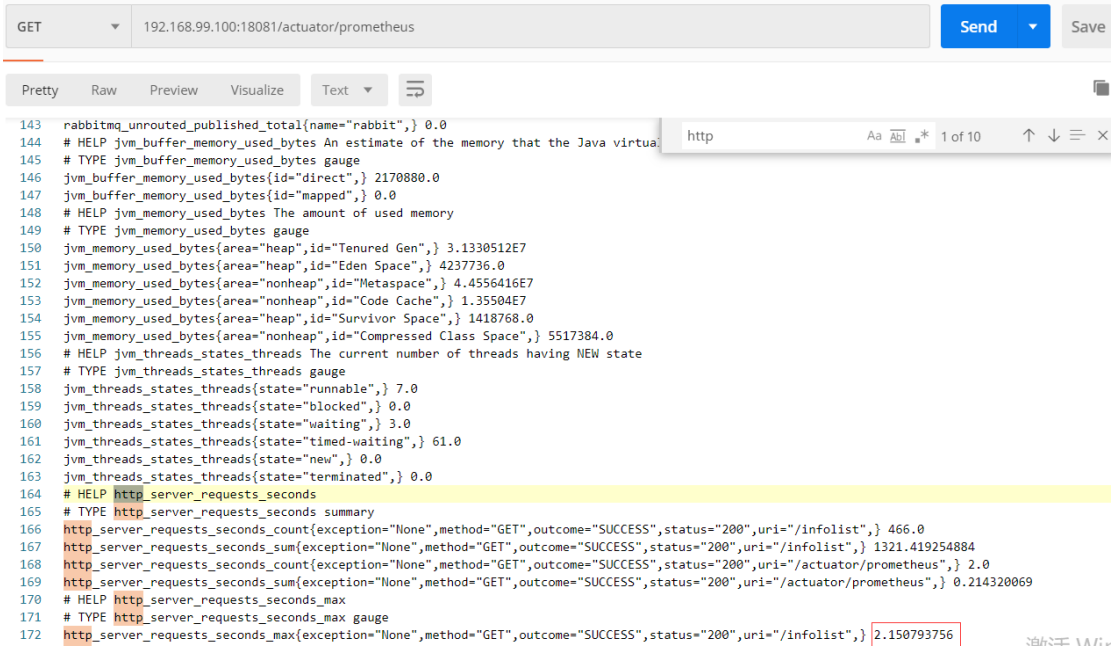
2.2 3个服务器

```
Simulation gatlingtest.GatlingTestSimulation completed in 2 seconds
Parsing log file(s)...
Parsing log file(s) done
Generating reports...

================================================================================
---- Global Information --------------------------------------------------------
> request count                                        400 (OK=400    KO=0     )
> min response time                                   1214 (OK=1214   KO=-     )
> max response time                                   2534 (OK=2534   KO=-     )
> mean response time                                  1939 (OK=1939   KO=-     )
> std deviation                                        348 (OK=348    KO=-     )
> response time 50th percentile                       1947 (OK=1947   KO=-     )
> response time 75th percentile                       2263 (OK=2263   KO=-     )
> response time 95th percentile                       2447 (OK=2447   KO=-     )
> response time 99th percentile                       2479 (OK=2479   KO=-     )
> mean requests/sec                                133.333 (OK=133.333 KO=-     )
---- Response Time Distribution ------------------------------------------------
> t < 800 ms                                            0 (  0%)
> 800 ms < t < 1200 ms                                  0 (  0%)
> t > 1200 ms                                         400 (100%)
> failed                                                0 (  0%)
================================================================================
```

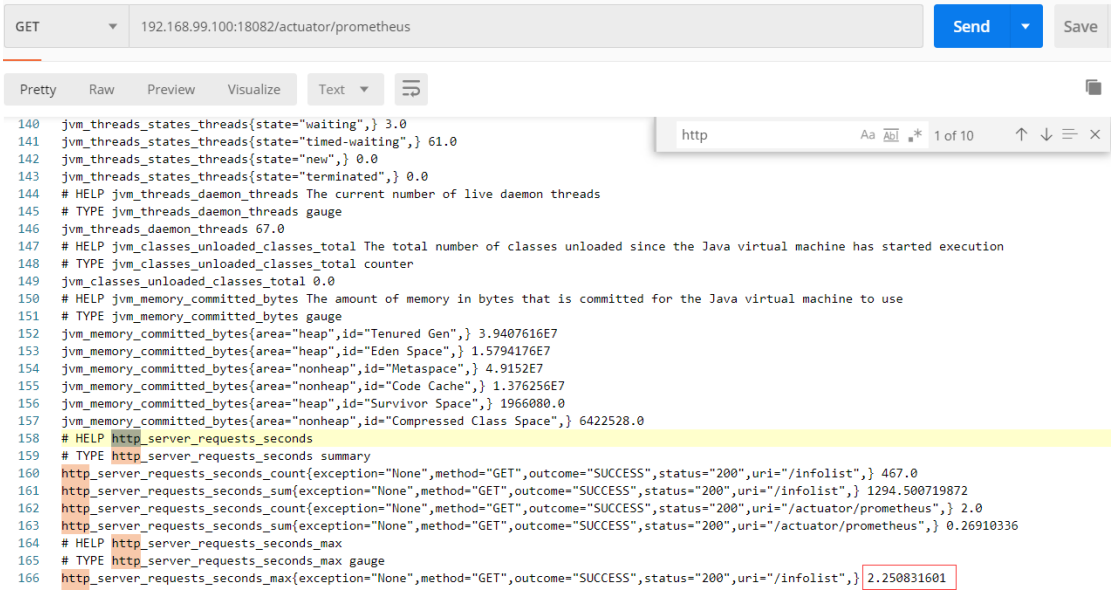
server0:



server1:



server2:



2.3 6个服务器

```
Simulation gatlingtest.GatlingTestSimulation completed in 3 seconds
Parsing log file(s)...
Parsing log file(s) done
Generating reports...

================================================================================
---- Global Information --------------------------------------------------------
> request count                                        400 (OK=400    KO=0     )
> min response time                                   1363 (OK=1363   KO=-     )
> max response time                                   3132 (OK=3132   KO=-     )
> mean response time                                  2364 (OK=2364   KO=-     )
> std deviation                                        502 (OK=502    KO=-     )
> response time 50th percentile                       2437 (OK=2437   KO=-     )
> response time 75th percentile                       2811 (OK=2811   KO=-     )
> response time 95th percentile                       3054 (OK=3054   KO=-     )
> response time 99th percentile                       3097 (OK=3097   KO=-     )
> mean requests/sec                                    100 (OK=100    KO=-     )
---- Response Time Distribution ------------------------------------------------
> t < 800 ms                                             0 (  0%)
> 800 ms < t < 1200 ms                                   0 (  0%)
> t > 1200 ms                                          400 (100%)
> failed                                                 0 (  0%)
================================================================================
```

server0：端口号 18080

```
# HELP http_server_requests_seconds
# TYPE http_server_requests_seconds summary
http_server_requests_seconds_count{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 1400.0
http_server_requests_seconds_sum{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 2670.6871294
http_server_requests_seconds_count{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/actuator/prometheus",} 6.0
http_server_requests_seconds_sum{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/actuator/prometheus",} 0.438210861
# HELP http_server_requests_seconds_max
# TYPE http_server_requests_seconds_max gauge
http_server_requests_seconds_max{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 1.946132594
```

server1：端口号 18081

```
# HELP http_server_requests_seconds
# TYPE http_server_requests_seconds summary
http_server_requests_seconds_count{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 599.0
http_server_requests_seconds_sum{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 1506.876000444
http_server_requests_seconds_count{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/actuator/prometheus",} 4.0
http_server_requests_seconds_sum{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/actuator/prometheus",} 0.355336322
# HELP http_server_requests_seconds_max
# TYPE http_server_requests_seconds_max gauge
http_server_requests_seconds_max{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 1.743644968
```

server2：端口号 18082

```
# HELP http_server_requests_seconds
# TYPE http_server_requests_seconds summary
http_server_requests_seconds_count{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 601.0
http_server_requests_seconds_sum{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 1482.059744733
http_server_requests_seconds_count{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/actuator/prometheus",} 4.0
http_server_requests_seconds_sum{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/actuator/prometheus",} 0.370624947
# HELP http_server_requests_seconds_max
# TYPE http_server_requests_seconds_max gauge
http_server_requests_seconds_max{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 1.880436662
```

server3：端口号 18083

```
# HELP http_server_requests_seconds
# TYPE http_server_requests_seconds summary
http_server_requests_seconds_count{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 200.0
http_server_requests_seconds_sum{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 979.304978893
http_server_requests_seconds_count{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/actuator/prometheus",} 1.0
http_server_requests_seconds_sum{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/actuator/prometheus",} 0.240837742
# HELP http_server_requests_seconds_max
# TYPE http_server_requests_seconds_max gauge
http_server_requests_seconds_max{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 2.822611694
```

server4：端口号 18084

```
# HELP http_server_requests_seconds
# TYPE http_server_requests_seconds summary
http_server_requests_seconds_count{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 200.0
http_server_requests_seconds_sum{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 952.59993955
# HELP http_server_requests_seconds_max
# TYPE http_server_requests_seconds_max gauge
http_server_requests_seconds_max{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 2.764066864
```

server5：端口号 18085

```
# HELP http_server_requests_seconds
# TYPE http_server_requests_seconds summary
http_server_requests_seconds_count{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 200.0
http_server_requests_seconds_sum{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 863.863954592
http_server_requests_seconds_count{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/actuator/prometheus",} 1.0
http_server_requests_seconds_sum{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/actuator/prometheus",} 0.18140711
# HELP http_server_requests_seconds_max
# TYPE http_server_requests_seconds_max gauge
http_server_requests_seconds_max{exception="None",method="GET",outcome="SUCCESS",status="200",uri="/infolist",} 2.893009913
```

三、测试结论

同样是 400 个并发的 http 请求，当只有一个服务器时，最小响应时间是 1948ms,最大响应时间是 4529ms；当有三个服务器时，最小响应时间 1214ms，最大响应时间 2534ms，比只有一个服务器时提速了不少；当有六个服务器时，最小响应时间 1363ms，最大响应时间 3132ms，与有三个服务器时相比，性能变化不大，说明服务器较多时，haproxy 调度消耗的时间也变多了。