KeenTune性能优化动手实践

Part 1. Profile的设置和回滚

Part 2. 参数智能优化

Part 3. 敏感参数识别

Step 1. 安装keentuned和keentune-target

yum install keentune-target, keentuned

KeenTune六大组件

1. Keentuned 主要控制组件
2. Keentune-target 参数设置组件
3. Keentune-brain 算法运行组件
4. Keentune-bench benchmark运行组件
5. Keentune-UI 前端组件
6. Keenopt 优化算法库

Step 2. 配置keentuned

vim /etc/keentune/conf/keentuned.conf

```
48 [target-group-1]
49 # * Topology of target group and knobs to be tuned in target. *#
50 # The machine ip address to depoly keentune-target.
51 TARGET_IP    = localhost
52 # The service port of keentune-target.
53 TARGET_PORT = 9873
54 # Knobs to be tuned in this target
55 PARAMETER    = sysctl.json, nginx_conf.json
```

Step 3. 启动服务，检查服务状态

- systemctl start keentune-target.service
- systemctl start keentuned.service
- systemctl status keentune-target.service
- systemctl status keentuned.service

```
● keentune-target.service - Parameter setting agent 'keentune-target' pertains to tuning tool 'KeenTune'.
   Loaded: loaded (/usr/lib/systemd/system/keentune-target.service; disabled; vendor preset: disabled)
   Active: active (running) since Thu 2023-06-08 11:32:33 CST; 1 day 4h ago
 Main PID: 86743 (keentune-target)
    Tasks: 1 (limit: 200527)
   Memory: 28.2M
   CGroup: /system.slice/keentune-target.service
           └─86743 /usr/libexec/platform-python /usr/bin/keentune-target

[root@keentune-linux-lug019 ~]# systemctl status keentuned.service
● keentuned.service - AI Tuning tool Daemon
   Loaded: loaded (/usr/lib/systemd/system/keentuned.service; disabled; vendor preset: disabled)
   Active: active (running) since Thu 2023-06-08 11:32:39 CST; 1 day 4h ago
 Main PID: 86799 (keentuned)
    Tasks: 14 (limit: 200527)
   Memory: 23.6M
   CGroup: /system.slice/keentuned.service
           └─86799 /usr/bin/keentuned
```

Step 4. 检查profile激活情况

keentune profile list

```
[root@keentune-linux-lug019 ~]# keentune profile list
- custom
- application
        [active]        nginx.conf      target_info: group1
        [available]     mysql.conf
        [available]     pgsql.conf
        [available]     redis.conf
- basic
        [available]     cpu_high_load.conf
        [available]     io_high_throughput.conf
        [available]     net_high_throuput.conf
        [available]     net_low_latency.conf
- ecs
        [available]     ecs-guest.conf
        [available]     ecs-performance.conf
```

KeenTune的profile自动设置功能：

Keentune启动后会自动识别当前运行环境，选择和设置当前环境最适用的profile，提升当前环境的性能表现

Step 5. 查看profile内容

keentune profile info --name virtual-guest.conf

```
[main]
summary=Optimize for running inside a virtual guest
include=throughput-performance

[sysctl]
# If a workload mostly uses anonymous memory and it hits this limit, the entire
# working set is buffered for I/O, and any more write buffering would require
# swapping, so it's time to throttle writes until I/O can catch up.  Workloads
# that mostly use file mappings may be able to use even higher values.
#
# The generator of dirty data starts writeback at this percentage (system default
# is 20%)
vm.dirty_ratio = 30

# Filesystem I/O is usually much more efficient than swapping, so try to keep
# swapping low.  It's usually safe to go even lower than this on systems with
# server-grade storage.
vm.swappiness = 30
```

Profile的继承关系：

通过“include”关键字，将另一个profile包含到当前profile当中，继承被包含profile的所有参数设置，可以在当前profile中重新定义参数值或增加参数。

Keentune中的参数域

Keentune将具有相同或相似设置方式的参数定义为一类参数域，例如“sysctl”和“nginx”

vm.dirty_ratio
- 处理缓存脏页的脏页占内存百分比阈值

vm.swappiness
- 开始使用交换分区的内存使用率阈值

Step 6. 检查基准性能指标

keentune profile rollback

KeenTune的回滚机制

KeenTune可以通过rollback命令将最近一次参数设置回滚到参数设置之前。

wrk -t 10 -c 300 -d 1 --latency http://127.0.0.1

```
Running 1s test @ http://127.0.0.1
  10 threads and 300 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency     1.04ms    1.54ms  24.58ms   89.61%
    Req/Sec    42.53k    15.70k   79.27k    68.87%
  Latency Distribution
     50%  468.00us
     75%    1.27ms
     90%    2.65ms
     99%    6.84ms
  449349 requests in 1.10s, 1.75GB read
Requests/sec: 408658.47
Transfer/sec:      1.60GB
```

wrk 性能测试工具

针对HTTP协议的单机、开源基准测试工具

Step 7. 设置nginx优化profile

keentune profile set nginx.conf

```
[root@keentune-linux-lug019 ~]# keentune profile set nginx.conf
[+] Recommendation (Manual Settings)
        [ssl]
                ssl_certificate:  Please use EDCSA instead of SHA256/SHA2048 or other certification methods.
                        Please refer to https://gitee.com/anolis/community/blob/master/sig/ARM_SIG/content/profile_fea
tures/SSL%E8%A7%A3%E5%AF%86%E7%AE%97%E6%B3%95EDCSA.md
        [kernel_sec]
                E0PD:  Please use E0PD security feature to avoid meltdown attack.
                        Please refer to https://gitee.com/anolis/community/blob/master/sig/ARM_SIG/content/profile_fea
tures/E0PD.md

        [WARNING] Settings in [vm.thunderx] domain only suites for #!uname_arch#= aarch64 & #!thunderx_cpu_info# = CPU
 part\s+:\s+(0x0?516)|(0x0?af)|(0x0?a[0-3])|(0x0?b8)\b Env, please set your parameters refer to throughput-performance
.conf
        [WARNING] Settings in [sysctl.thunderx] domain only suites for #!uname_arch#= aarch64 & #!thunderx_cpu_info# =
 CPU part\s+:\s+(0x0?516)|(0x0?af)|(0x0?a[0-3])|(0x0?b8)\b Env, please set your parameters refer to throughput-perform
ance.conf
        [WARNING] [sysctl] net.ipv4.tcp_tw_timeout: 'net.ipv4.tcp_tw_timeout' can not backup
        [WARNING] [cpu] force_latency: 'force_latency' can not backup
        [WARNING] [cpu] governor: 'governor' can not backup
        [WARNING] [cpu] min_perf_pct: 'min_perf_pct' can not backup
        [WARNING] unvaliable domain 'disk': No device is avaliable in this environment.

[+] Profile Result (Auto Settings)
        Group 1 Node 1: localhost
                [limits] 4 Succeeded, 0 Failed, 0 Skipped
                [net] 3 Succeeded, 0 Failed, 0 Skipped
                [nginx_conf] 14 Succeeded, 0 Failed, 0 Skipped
                [sysctl] 18 Succeeded, 0 Failed, 0 Skipped
```

KeenTune参数设置的三种结果

Recommendation
推荐用户手动设置，不宜自动设置的参数

Warning
当前环境不适用的参数，不影响其他参数设置

Succeeded
成功设置的参数

Step 8. 检查优化效果

```
wrk -t 10 -c 300 -d 1 --latency http://127.0.0.1
```

```
Running 1s test @ http://127.0.0.1
  10 threads and 300 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency     1.04ms    1.54ms  24.58ms   89.61%
    Req/Sec    42.53k    15.70k   79.27k    68.87%
  Latency Distribution
     50%  468.00us
     75%    1.27ms
     90%    2.65ms
     99%    6.84ms
  449349 requests in 1.10s, 1.75GB read
Requests/sec: 408658.47
Transfer/sec:      1.60GB
```

优化前

```
Running 1s test @ http://127.0.0.1
  10 threads and 300 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency     1.57ms    3.47ms  52.84ms   90.77%
    Req/Sec    54.20k    37.90k  137.47k    56.07%
  Latency Distribution
     50%  346.00us
     75%    1.22ms
     90%    4.68ms
     99%   12.85ms
  578909 requests in 1.11s, 2.26GB read
Requests/sec: 520960.48
Transfer/sec:      2.03GB
```

优化后

Step 9. 通过网页控制和查看结果

http:// localhost:8082/home

KeenTune性能优化动手实践

Part 1. Profile的设置和回滚

👉 Part 2. 参数智能优化

Part 3. 敏感参数识别

Step 1. 安装keentune-brain和keentune-bench

- yum install keentune-bench
- yum install keentune-brain

KeenTune六大组件

1. Keentuned 主要控制组件
2. Keentune-target 参数设置组件
3. Keentune-brain 算法运行组件
4. Keentune-bench benchmark运行组件
5. Keentune-UI 前端组件
6. Keenopt 优化算法库

Step 2. 配置参数智能优化算法

vim /etc/keentune/conf/keentuned.conf

```
37 [brain]
38 # * Topology of brain and basic configuration about brain. *#
39 # The machine ip address to depoly keentune-brain.
40 BRAIN_IP                = localhost
41 # The service port of keentune-brain
42 BRAIN_PORT              = 9872
43 # Brain optimization algorithm. i.e. tpe, hord, random
44 AUTO_TUNING_ALGORITHM = hord
45 # Explainer of sensitive parameter training. i.e. shap, lasso, univariate
46 SENSITIZE_ALGORITHM    = shap
```

Brain所在ip地址

Brain运行的参数智能优化算法

HORD

Hyperparameter Optimization using RBF based surrogate and DYCORS

一种基于RBF插值函数和变异搜索的序列优化算法

Step 3. 配置benchmark脚本

vim /etc/keentune/conf/keentuned.conf

```
57 [bench-group-1]
58 # * Topology of bench group and benchmark script to be performed in bench. *#
59 # The machine ip address to depoly keentune-bench.
60 BENCH_SRC_IP    = localhost
61 # The service port of keentune-bench.
62 BENCH_SRC_PORT = 9874
63 # The destination ip address in benchmark workload.
64 BENCH_DEST_IP   = localhost
65 # The configuration file of benchmark to be performed
66 BENCH_CONFIG    = wrk_http_long.json
```

Bench所在ip地址

Bench上运行的benchmark脚本

wrk 性能测试工具

针对HTTP协议的单机、开源基准测试工具

Step 4. 配置待调优参数

vim /etc/keentune/conf/keentuned.conf

```
48 [target-group-1]
49 # * Topology of target group and knobs to be tuned in target. *#
50 # The machine ip address to depoly keentune-target.
51 TARGET_IP    = localhost
52 # The service port of keentune-target.
53 TARGET_PORT = 9873
54 # Knobs to be tuned in this target
55 PARAMETER    = sysctl.json, nginx_conf.json
```

Target所在ip地址

待调优参数列表

KeenTune多参数联合调优

针对多个参数域同时进行调优，有效扩大了参数组合的搜索范围，有利于找到更好的参数配置。

Step 5. 设置参数优化目标

vim /etc/keentune/benchmark/wrk/wrk_http_long.json

```
{
  "benchmark": [
    {
      "benchmark_cmd": "python3 {remote_script_path} {target_ip}",
      "local_script_path": "benchmark/wrk/wrk_http_long.py",
      "items": {
        "Requests_sec": {
          "negative": false,
          "weight": 100,
          "strict": false
        },
        "Transfer_sec": {
          "negative": false,
          "weight": 0,
          "strict": false
        },
        "Latency_90": {
          "negative": true,
          "weight": 0,
          "strict": true
        },
        "Latency_99": {
          "negative": true,
          "weight": 0,
          "strict": true
```

KeenTune多目标调优

通过配置多个优化目标的权重比例，使优化向着不同的方向进行，可以使优化参数配置兼顾多个性能指标。

“Strict”模式

避免多目标调优中可能会出现的一个性能指标优化，另一个性能指标变差的问题

Step 6. 运行参数智能优化

keentune param tune --job test --iteration 10

```
[root@keentune-linux-lug019 ~]# keentune param tune --job test --iteration 10
[ok] Running Param Tune Success.

        iteration: 10
        name: test

        see more details by log file: "/var/log/keentune/test.log"
```

Step 7. 查看参数智能优化结果

cat /var/log/keentune/test.log

```
[Iteration 9] Benchmark result:
        [Requests_sec]  (weight: 100.0) average scores = 443068.594
        Current optimal iteration: [Iteration 6]:
        [Requests_sec]  Improved by 12.708%
[Iteration 10] Benchmark result:
        [Requests_sec]  (weight: 100.0) average scores = 521366.438
        Current optimal iteration: [Iteration 10]:
        [Requests_sec]  Improved by 28.986%

Step5. Best configuration dump successfully. File list:
        /var/keentune/tuning_workspace/test/test_group1_best.json


Step6. Tuning is finished, checking benchmark score of best configuration.

[BEST] Benchmark result:
        [Requests_sec]  (weight: 100.0) average scores = 516432.938
[BEST] Tuning improvement:
        [Requests_sec]  Improved by 27.766%
```

KeenTune重复检验和优化后检验

调优中每轮调优会多次运行

调优完成之后设置最优参数配置，执行多次benchmark减少随机干扰，确认优化效果。

Step 8. 通过网页控制和查看结果

http://localhost:8082/home

KeenTune性能优化动手实践

Part 1. Profile的设置和回滚

Part 2. 参数智能优化

👉 Part 3. 敏感参数识别

Step 1. 配置敏感参数算法

vim /etc/keentune/conf/keentuned.conf

```
[brain]
# * Topology of brain and basic configuration about brain. *#
# The machine ip address to depoly keentune-brain.
BRAIN_IP               = localhost
# The service port of keentune-brain
BRAIN_PORT             = 9872
# Brain optimization algorithm. i.e. tpe, hord, random
AUTO_TUNING_ALGORITHM = hord
# Explainer of sensitive parameter training. i.e. shap, lasso, univariate
SENSITIZE_ALGORITHM    = shap
```

Step 2. 选择数据集

keentune param jobs

```
[root@keentune-linux-lug019 ~]# keentune param jobs
name    algorithm       iteration       status  start_time      end_time
demo    random  200     finish  2023-06-08 16:51:59     2023-06-08 17:00:53
test    hord    10      finish  2023-06-09 16:56:03     2023-06-09 16:56:38
```

KeenTune敏感参数识别数据集

敏感参数识别通过分析不同参数组合下的性能表现来识别参数的重要性，KeenTune将参数智能优化的数据直接用于敏感参数识别，但最好使用“random“等采样算法，使数据没有偏向性

Step 3. 执行敏感参数识别算法

keentune sensitize train --data demo --job demo --trials 5

cat /var/log/keentune/keentuned-sensitize-train-demo.log

param name	round 1	round 2	round 3	round 4	round 5
sendfile@group-1	-0.0001	0.0004	-0.0000	0.0000	-0.0002
tcp_nopush@group-1	-0.0007	0.0001	0.0002	0.0001	-0.0001
vm.min_free_kbytes@group-1	0.0028	0.0174	-0.0004	0.0085	0.0051
proxy_read_timeout@group-1	-0.0001	0.0005	0.0001	-0.0041	0.0002
kernel.msgmnb@group-1	-0.0000	0.0001	-0.0000	0.0000	-0.0000
net.ipv4.tcp_dsack@group-1	0.0007	-0.0000	0.0000	0.0002	0.0087
net.ipv4.tcp_orphan_retries@group-1	-0.0230	0.0001	0.0000	0.0000	0.0001
net.ipv4.tcp_max_tw_buckets@group-1	0.0000	0.0009	-0.0000	-0.0004	-0.0000
net.ipv4.tcp_max_syn_backlog@group-1	-0.0001	0.0000	0.0000	-0.0001	0.0000
net.ipv4.tcp_fin_timeout@group-1	-0.0002	0.0001	0.0001	0.0000	0.0000
client_body_timeout@group-1	-0.0001	0.0002	0.0063	-0.0000	0.0012
net.ipv4.tcp_keepalive_probes@group-1	-0.0003	-0.0000	-0.0000	-0.0001	-0.0000
net.core.somaxconn@group-1	0.0000	0.0000	0.0000	0.0005	0.0000
kernel.msgmni@group-1	0.0000	-0.0001	0.0000	0.0001	0.0001

KeenTune敏感参数识别的偶然性

敏感参数识别算法存在一定的随机性，每次运行的结果不完全一致，所以通过多次运行来减少偶然性

KeenTune参数敏感性的含义

参数敏感性结果是 -1～1 之间的值，敏感值为正表示与性能指标呈正相关，为负表示与性能指标呈负相关，绝对值越大影响程度越大

Step 4. 通过网页控制和查看结果

http:// localhost :8082/home