**HTP MySQL审计使用手册**

**上海热璞网络科技技术有限公司**

**DRDS 组**

2018年10月

# **文档版本**

|  |  |  |  |
| --- | --- | --- | --- |
| **版本号** | **编写人** | **编写日期** | **更改内容** |
| 1.0 | 姜宇祥 | 2015-2-15 | 创建文档 |
| 1.1 | 姜宇祥 | 2015-3-24 | 修改变量说明 |
| 1.1 | 姜宇祥 | 2015-4-2 | 根据程序功能变化，增加相应的异常说明 |
| 1.1 | 姜宇祥 | 2015-4-3 | 增加状态的说明 |
| 1.2 | 郝国庆 | 2018-10-17 | 修改配置文件格式 |
| 1.2 | 郝国庆 | 2018-10-24 | 补全5.7之后增加的事件审计 |
|  |  |  |  |
|  |  |  |  |

# **目录**

**[文档版本](#_Toc530518393)** [2](#_Toc530518393)

[1.0 2](#_Toc530518394)

**[目录](#_Toc530518395)** [3](#_Toc530518395)

**[一、引言](#_Toc530518396)** [4](#_Toc530518396)

[1编写目的 4](#_Toc530518397)

[2参考资料 4](#_Toc530518398)

[3术语和缩写词 4](#_Toc530518399)

[二、概述 5](#_Toc530518400)

[1、用途 5](#_Toc530518401)

[2、运行环境 5](#_Toc530518402)

[3、性能 5](#_Toc530518403)

[三、使用 6](#_Toc530518404)

[1、构成 6](#_Toc530518405)

[2、安装 6](#_Toc530518406)

[2-1加载插件 6](#_Toc530518407)

[2-2卸载插件 6](#_Toc530518408)

[3、使用配置文件 6](#_Toc530518409)

[3-1、配置文件的格式 6](#_Toc530518410)

[3-2、配置 7](#_Toc530518411)

[4、运行时变量 11](#_Toc530518412)

[4-1、变量 11](#_Toc530518413)

[4-2、变量说明 12](#_Toc530518414)

[5、运行时状态 14](#_Toc530518415)

[6、输出 17](#_Toc530518416)

[7、异常 19](#_Toc530518417)

# **一、引言**

## 1编写目的

编写本使用说明的目的是充分叙述本软件所能实现的功能及其运行环境，以便使用者了解本软件的使用范围和使用方法，并为软件的维护和更新提供必要的信息。

## 2参考资料

1、MySQL Enterprise Audit Log Plugin

https://dev.mysql.com/doc/refman/5.7/en/audit-log-logging-control.html

## 3术语和缩写词

审计插件：如无特殊说明，特指热璞的MySQL审计插件

# 二、概述

## 1、用途

用于MySQL数据库的事件审计

## 2、运行环境

本软件作为插件运行在热璞定制的MySQL 5.7.18版本上，MySQL的工作环境为CentOS 6/7

## 3、性能

在全部审计功能打开的情况下，MySQL性能下降不超过10%

# 三、使用

## 1、构成

提供源码安装。在plugin 目录下新建目录htp\_audit，将源码文件拷贝至该目录，进行编译，编译安装后会生成动态链接库hotpu\_audit.so。配置文件hotpu\_audit.cnf 参照模板。

## 2、安装

### 2-1加载插件

install plugin htp\_audit soname 'htp\_audit.so';

### 2-2卸载插件

uninstall plugin htp\_audit;

## 3、使用配置文件

热璞MySQL审计插件使用独立的配置文件，并不从MySQL的配置文件读取配置信息。通过读取配置文件获得运行时所需信息，并根据这些配置信息完成相应的审计功能。

审计插件的配置文件需要与插件放在相同目录，也就是MySQL的插件目录中。如果没有找到配置文件，审计插件将按照默认模式进行工作。

配置文件由通用组和筛选组构成。通用组用于配置审计插件的通用属性，筛选组用于配置运行时的筛选规则，每一个筛选组对应一个筛选规则。

### 3-1、配置文件的格式

审计插件是按照行进行配置读取的，每一行代表一个完整的内容，不支持分行的配置内容读取，每一行称为配置行。

配置行分为如下4类，配置组名，配置项，注释和空行。其中注释是以“#”字符开头的行，空行是包含空格/制表符/回车/换行字符，这两种行将被审计插件忽略。配置组名和配置项将在后面说明。

### 3-2、配置

审计插件支持两种配置组。通用配置组和筛选配置组。

通用配置组须以[general]为内容的配置行开始，其后的为通用配置的配置项。

筛选配置组须以[audit rule]为内容的配置行开始，其后为筛选组的配置内容。

#### 3-2-1通用配置组

通用配置组支持的配置为

log\_file:用于记录审计信息

err\_file:用于记录用户请求执行时出现的错误

enable\_buffer:审计时是否使用缓冲区。使用缓冲区将会提高程序的运行效率，但发生宕机情况时会出现部分数据未写入磁盘，导致部分审计信息丢失的情况

#### 3-2-2筛选配置组

配置说明

筛选配置组支持的配置为

name:筛选配置的名字

user:进行筛选的用户名字

host:进行筛选的主机名或ip

event:进行筛选的事件的指定，被指定的事件将会被审计，其他的事件将会被忽略。事件之间通过分号（;）进行分隔

热璞审计的事件分为如下类，

|  |  |
| --- | --- |
| 审计类名称 | 说明 |
| CONNECTION | 连接类型事件 |
| GENERAL | 普通类型事件 |
| PARSE | 解析类型事件 |
| AUTHORIZATION | 授权类型事件 |
| TABLE ACCESS | 表访问类型事件 |
| GLOBAL VARIABLE | 全局变量类型事件 |
| COMMAND | 命令类型事件 |
| QUERY | 查询类型事件 |
| SERVER STARTUP | 实例启动类型事件 |
| SERVER SHUTDOWN | 实例关闭类型事件 |
| STORED PROGRAM | 存储程序类型事件 |

每个大类包含如下小类

CONNECTION：

|  |  |
| --- | --- |
| 子类名称 | 说明 |
| CONNECT | 建连事件 |
| DISCONNECT | 断连事件 |
| CHANGE USER | 改变用户 |
| PRE AUTHENTICATE | 预授权事件 |

GENERAL：

|  |  |
| --- | --- |
| 子类名称 | 说明 |
| LOG | 日志事件 |
| ERROR | 错误事件 |
| RESULT | 结果事件 |
| STATUS | 状态事件 |

PARSE：

|  |  |
| --- | --- |
| 子类名称 | 说明 |
| PREPARSE | 预解析事件 |
| POSTPARE | 解析完成事件 |

AUTHORIZATION：

|  |  |
| --- | --- |
| 子类名称 | 说明 |
| USER | 用户授权事件 |
| DB | 数据库授权事件 |
| TABLE | 表授权事件 |
| COLUMN | 列授权事件 |
| PROCEDURE | 存储过程授权事件 |
| PROXY | 代理授权事件 |

TABLE ACCESS：

|  |  |
| --- | --- |
| 子类名称 | 说明 |
| READ | 读表事件 |
| INSERT | 插入事件 |
| UPDATE | 更新时间 |
| DELETE | 删除时间 |

GLOBAL VARIABLE：

|  |  |
| --- | --- |
| 子类名称 | 说明 |
| SET | 设置变量事件 |
| GET | 读取变量事件 |

SERVER STARTUP：

|  |  |
| --- | --- |
| 子类名称 | 说明 |
| STARTUP | 启动事件 |

SERVER SHUTDOWN

|  |  |
| --- | --- |
| 子类名称 | 说明 |
| SHUTDOWN | 关闭事件 |

COMMAND：

|  |  |
| --- | --- |
| 子类名称 | 说明 |
| START | 命令开始事件 |
| END | 命令结束事件 |

QUERY：

|  |  |
| --- | --- |
| 子类名称 | 说明 |
| START | 查询开始事件 |
| NESTED START | 嵌套查询开始事件 |
| END | 查询结束事件 |
| NESTED END | 嵌套查询结束事件 |

STORED PROGRAM:

|  |  |
| --- | --- |
| 子类名称 | 说明 |
| STORED\_PROGRAM\_EXECUTE | 存储程序执行事件 |

#### 3-2-3书写要求

1、除空配置行外，其他配置行不接受以空格为开始的配置行。

2、配置组/配置项不区分大小写。

3、配置项以key-value方式书写，’=’字符作为分隔符，前端为key，后端为value，’=’两端的空格也作为key和value的输入

4、筛选组中，用户/主机名配置项中value允许使用的字符只限定在如下范围：a到z，A到Z，0到9，\_和.等字符，不能使用其他字符。

#### 3-2-4 配置项说明

|  |  |
| --- | --- |
| 配置项名称 | 说明 |
| name | 配置筛选规则的名称 |
| user | 进行筛选的登录账号名称 |
| host | 进行筛选的主机名或ip地址 |
| event | 进行审计的事件类型配置，事件之间用’;’分割，子事件之间用’,’号分隔 |
| command | 进行审计的命令类型(可用值：query,execute) |
| sql\_command | 进行审计的sql语句类型 |
| sql\_keyword | 进行审计的sql关键词 |

其中，**command,sql\_command,sql\_keyword** 仅在 event 含有general的时候有效，sql\_command 在command 设置为query或execute时，可以设置为如下内容：alter\_db,alter\_Db\_upgrade,alter\_event等(更多类型可以通过SELECT NAME FROM performance\_schema.setup\_instruments WHERE NAME LIKE 'statement/sql/%' ORDER BY NAME 获取)。

#### 3-2-5样例

[general]

audit\_file=htp\_audit.log

audit\_error\_file=htp\_audit\_error.log

enable\_buffer=1

[audit rule]

name=rule1

user=root

host=192.168

event={connection:connect,disconnect};{general:status,error}

command=query

sql\_command=set\_option

sql\_keyword=names

[audit rule]

name=rule2

user=root

host=192.168

event={shutdown};{table access:read,insert,update};{startup};{stored program}

command = query

sql\_command=set\_option

sql\_keyword=names

#### 3-2-6异常说明

在读取/分析/使用配置文件时，一旦发生异常，hotpu的审计插件通过msyql的error log文件进行错误信息的记录。

##### 可能的异常情况

1、当配置文件中出现重名筛选配置组

2、在配置组中重复指定配置项

##### 错误信息样例

2015-04-02 15:53:49 60735 [ERROR] Hotpu Audit: duplicate user in one rule setting

2015-04-02 15:57:48 60735 [ERROR] Hotpu Audit: group 1 error

## 4、运行时变量

通过查询一些运行时变量，用户可以了解审计运行中的参数；通过设置一些运行时变量，改变审计运行中的参数，影响审计的行为。

### 4-1、变量

审计插件所使用的变量如下表所列

|  |  |  |
| --- | --- | --- |
| Name | Option File | Dynamic |
| htp\_audit\_add\_rule |  | Yes |
| htp\_audit\_buffer\_size |  | Yes |
| htp\_audit\_enable\_buffer | Yes | Yes |
| htp\_audit\_error\_log\_file | Yes | No |
| htp\_audit\_flush\_log |  | Yes |
| htp\_audit\_log\_file | Yes | No |
| htp\_audit\_remove\_rule |  | Yes |
| htp\_audit\_rules |  | No |

### 4-2、变量说明

#### htp\_audit\_add\_rule

通过设置该变量，增加审计筛选条件。增加的筛选条件的名字必须不同于当前审计插件的任意一个筛选条件的名字，否则会报告错误。

|  |  |  |
| --- | --- | --- |
| Introduced | 1.1 | |
| Permitted Values | Type | string |

样例:

set global htp\_audit\_add\_rule='name=rule2;event={parse};{general:log,error};host=localhost;user=user1';

#### hotpu\_audit\_buffer\_size

设置缓冲区的大小，以KB为单位进行设置。如1表示1Kb，也就是1024个字节。当32时，会自动设置为32。

|  |  |  |
| --- | --- | --- |
| Introduced | 1.1 | |
| Permitted Values | Type | integer |
| Default | 32 |
| Unit | Kb |
| Min Value | 32 |
| Max Value | 1024 \* 4 |

样例

set global htp\_audit\_buffer\_size=32;

#### htp\_audit\_enable\_buffer

打开审计缓冲功能

|  |  |  |
| --- | --- | --- |
| Introduced | 1.1 | |
| Permitted Values | Type | Boolean |
| Default | false |

样例

set global htp\_audit\_enable\_buffer=true;

#### htp\_audit\_error\_log\_file

通过该变量展示保存发生错误的审计信息内容的文件

|  |  |  |
| --- | --- | --- |
| Introduced | 1.1 | |
| Permitted Values | Type | string |

样例

只读变量，只能通过show variable得到当前的信息

#### htp\_audit\_flush\_log

刷新当前的的审计文件，旧的数据将从当前的文件中删除，保存在其他文件中。新的数据继续写入当前的文件。

该变量的值只能被设置为true，用以表示进行审计信息的刷新，设置其他值将会报错。在完成设置之后的一系列操作，该变量自动被设置为false。

|  |  |  |
| --- | --- | --- |
| Introduced | 1.1 | |
| Permitted Values | Type | Boolean |
| Default | false |

样例

set global htp\_audit\_flush\_log=true;

#### htp\_audit\_log\_file

通过该变量展示保存审计信息内容的文件

|  |  |  |
| --- | --- | --- |
| Introduced | 1.1 | |
| Permitted Values | Type | string |

样例

只读变量，只能通过show variable得到当前的信息

#### htp\_audit\_remove\_rule

移除审计的筛选条件。通过指定筛选条件的名字，进行筛选条件的删除

|  |  |  |
| --- | --- | --- |
| Introduced | 1.1 | |
| Permitted Values | Type | string |

样例

set global htp\_audit\_remove\_rule='name=rule2';

#### htp\_audit\_rules

通过该变量展示当前有效的审计筛选条件

|  |  |  |
| --- | --- | --- |
| Introduced | 1.1 | |
| Permitted Values | Type | string |

样例

只读变量，只能通过show variable得到当前的信息

#### htp\_audit\_version

通过该变量展示当前审计插件的版本

|  |  |  |
| --- | --- | --- |
| Introduced | 1.1 | |
| Permitted Values | Type | string |

样例

只读变量，只能通过show variable得到当前的信息

## 5、运行时状态

通过查询运行时的状态可以了解热璞审计插件的运行状态，支持的状态包括如下。

|  |  |
| --- | --- |
| 状态变量 | 说明 |
| htp\_audit\_called | 调用审计插件的次数 |
| htp\_audit\_authorization\_column\_called | authonrization column 事件发生次数 |
| htp\_audit\_authorization\_db\_called | authonrization db事件发生次数 |
| htp\_audit\_authorization\_procedure\_called | authonrization procedure 发生次数 |
| htp\_audit\_authorization\_proxy\_called | authonrization proxy 发生次数 |
| htp\_audit\_authorization\_table\_called | authonrization table 发生次数 |
| htp\_audit\_authorization\_user\_called | authonrization user 事件发生次数 |
| htp\_audit\_command\_end\_called | command end 事件发生次数 |
| htp\_audit\_command\_start\_called | command start 事件发生次数 |
| htp\_audit\_connection\_change\_user\_called | connection change user 事件发生次数 |
| htp\_audit\_connection\_connect\_called | connection connect 事件发生次数 |
| htp\_audit\_connection\_disconnect\_called | connection disconnect 事件发生次数 |
| htp\_audit\_general\_error\_called | general error 事件发生次数 |
| htp\_audit\_general\_log\_called | general log 事件发生次数 |
| htp\_audit\_general\_result\_called | general result 事件发生次数 |
| htp\_audit\_general\_status\_called | general status事件发生次数 |
| htp\_audit\_global\_variable\_get\_called | global variable get 事件发生次数 |
| htp\_audit\_global\_variable\_set\_called | global variable set 事件发生次数 |
| htp\_audit\_parse\_postparse\_called | parse postparse 事件发生次数 |
| htp\_audit\_parse\_preparse\_called | parse preparse 事件发生次数 |
| htp\_audit\_query\_nested\_start\_called | query nested start 事件发生次数 |
| htp\_audit\_query\_nested\_status\_end\_called | query nested end 事件发生次数 |
| htp\_audit\_query\_start\_called | query start 事件发生次数 |
| htp\_audit\_query\_status\_end\_called | query end事件发生次数 |
| htp\_audit\_table\_access\_delete\_called | table access delete 事件发生次数 |
| htp\_audit\_table\_access\_insert\_called | table access insert 事件发生次数 |
| htp\_audit\_table\_access\_read\_called | table access read事件发生次数 |
| htp\_audit\_table\_access\_update\_called | table access update事件发生次数 |
| Htp\_audit\_stored\_program\_called | 存储程序执行事件发生次数 |
| htp\_audit\_recorded | 进行记录的事件的次数 |
| htp\_audit\_authorization\_column\_recorded | authonrization column 事件记录次数 |
| htp\_audit\_authorization\_db\_recorded | authonrization db事件记录次数 |
| htp\_audit\_authorization\_procedure\_recorded | authonrization procedure 记录次数 |
| htp\_audit\_authorization\_proxy\_recorded | authonrization proxy 事件记录次数 |
| htp\_audit\_authorization\_table\_recorded | authonrization table事件记录次数 |
| htp\_audit\_authorization\_user\_recorded | authonrization user 事件记录次数 |
| htp\_audit\_command\_end\_recorded | command end 事件审计次数 |
| htp\_audit\_command\_start\_recorded | command start 事件审计次数 |
| htp\_audit\_connection\_change\_user\_recorded | connection change user 事件审计次数 |
| htp\_audit\_connection\_connect\_recorded | connection connect 事件审计次数 |
| htp\_audit\_connection\_disconnect\_recorded | connection disconnect 事件审计次数 |
| htp\_audit\_general\_error\_recorded | general error 事件审计次数 |
| htp\_audit\_general\_log\_recorded | general log 事件审计次数 |
| htp\_audit\_general\_result\_recorded | general result 事件审计次数 |
| htp\_audit\_general\_status\_recorded | general status事件审计次数 |
| htp\_audit\_global\_variable\_get\_recorded | global variable get 事件审计次数 |
| htp\_audit\_global\_variable\_set\_recorded | global variable set 事件审计次数 |
| htp\_audit\_parse\_postparse\_recorded | parse postparse 事件审计次数 |
| htp\_audit\_parse\_preparse\_recorded | parse preparse 事件审计次数 |
| htp\_audit\_query\_nested\_start\_recorded | query nested start 事件记录次数 |
| htp\_audit\_query\_nested\_status\_end\_recorded | query nested end 事件审计次数 |
| htp\_audit\_query\_start\_recorded | query start 事件审计次数 |
| htp\_audit\_query\_status\_end\_recorded | query end事件记录次数 |
| htp\_audit\_table\_access\_delete\_recorded | table access delete 事件审计次数 |
| htp\_audit\_table\_access\_insert\_recorded | table access insert 事件审计次数 |
| htp\_audit\_table\_access\_read\_recorded | table access read事件审计次数 |
| htp\_audit\_table\_access\_update\_recorded | table access update事件审计次数 |
| Htp\_audit\_stored\_program\_recorded | 存储程序执行的审计次数 |

注：指定类型事件如果发生调用计数+1，如果在配置文件中配置该事件，则相应审计技术+1，否则无变化。

## 6、输出

审计事件以json的数据格式输出到审计文件中。如下样例所示

样例1

general类型的审计结果

{

"timestamp": "2015-03-25 12:22:56",

"type": "general",

"user": "root[root] @ localhost []",

"host": "localhost",

"ip": "",

"command\_class": "select",

"sqltext": "select 1",

"code": 0

}

样例2

connection类型的审计结果

{

"timestamp": "2015-03-25 11:56:42",

"type": "connection",

"connection type": "connect",

"host": "localhost",

"ip": "",

"user": "root"

}

样例3

parse 类型的审计结果

{

"timestamp": "2018-10-19 10:50:16",

"type": "parse",

"sub type": "preparse",

"sqltext": "show global variables like '%audit%'",

"pluginflag": 0

}

样例4

table access 类型的审计结果

{

"timestamp": "2018-10-19 16:42:36",

"type": "table access",

"sub type": "read",

"connection\_id": 3,

"sql\_command\_id": 0,

"sqltext": "select \* from audit\_test1",

"table\_database": "audit\_testdb",

"table\_name": "audit\_test1"

}

样例 5

global variable 类型审计结果

{

"timestamp": "2018-10-19 18:08:46",

"type": "global\_variable",

"sub type": "get",

"connection\_id": 3,

"sql\_command\_id": 31,

"variable\_name": "sync\_binlog",

"variable\_value": "0"

}

样例 6

command 类型审计结果

{

"timestamp": "2018-10-19 18:08:46",

"type": "command",

"sub type": "start",

"status": 0,

"connectionid": 1,

"commandid": 3

}

样例7

query 类型审计结果

{

"timestamp": "2018-10-21 15:24:55",

"type": "query",

"sub type": "start",

"status": 0,

"connectionid": 3,

"sqlcommandid": 118,

"sqltext": "show plugins"

}

样例8

stored program 类型审计结果

{

"timestamp": "2018-10-29 15:52:37",

"type": "stored program",

"sub type": "execute",

"connection\_id": 3,

"sql\_command\_id": 91,

"query": "call my\_proc",

"name": "my\_proc"

}

## 7、异常

1、配置文件相关异常尝试阅读配置文件一章。

2、在设置运行时变量未能成功的情况下，报告错误，并保持设置命令之前的状态

3、连接在特定情况下出现审计失效的问题。在连接非常活跃的情况，由于MySQL的实现机制会导致连接的资源无法被正确释放。

如下是MySQL出现问题的代码。红色标注部分是错误的核心，在插件卸载后，连接的工作线程执行到该函数的红色部分跳出，导致后续的释放代码没有正确执行。其中标志位没有重置（蓝色标注部分），导致下次进入审计时，无法正确的判断连接状态。

void mysql\_audit\_release(THD \*thd)

{

plugin\_ref \*plugins, \*plugins\_last;

if (!thd || !(thd->audit\_class\_plugins.elements))

return;

plugins= (plugin\_ref\*) thd->audit\_class\_plugins.buffer;

plugins\_last= plugins + thd->audit\_class\_plugins.elements;

for (; plugins < plugins\_last; plugins++)

{

st\_mysql\_audit \*data= plugin\_data(\*plugins, struct st\_mysql\_audit \*);

/\* Check to see if the plugin has a release method \*/

if (!(data->release\_thd))

continue;

/\* Tell the plugin to release its resources \*/

data->release\_thd(thd);

}

/\* Now we actually unlock the plugins \*/

plugin\_unlock\_list(NULL, (plugin\_ref\*) thd->audit\_class\_plugins.buffer,

thd->audit\_class\_plugins.elements);

/\* Reset the state of thread values \*/

reset\_dynamic(&thd->audit\_class\_plugins);

memset(thd->audit\_class\_mask, 0, sizeof(thd->audit\_class\_mask));

}