TiDB Audit Plugin User Guide

August 4, 2022

Introduction

The TiDB audit plugin records the TiDB server's activities that are expected to follow auditing regulations of your organization. For each client session, the plugin records users accessing the server (that is, username and IP address), SQL queries executed, databases and tables accessed, environment variable changes, and DDL operations executed. The auditing information is stored in a log file for future use.

This document describes how to compile, package, and use the audit plugin.

Download the plugin

You can download the plugin on TiDB Enterprise Edition Downloads.

Deploy the audit plugin

After downloading the plugin, you can use either TiDB Operator or TiUP to deploy the audit plugin.

Use TiDB Operator to deploy the plugin

Configure TidbCluster CR.

```
tidb:
  additionalContainers:
  - command:
    - sh
    - -c
    - touch /var/log/tidb/tidb-audit.log; tail -n0 -F /var/log/tidb/tidb-audit.log;
    image: busybox:1.26.2
```

```
  imagePullPolicy: IfNotPresent
  name: auditlog
  resources:
    limits:
      cpu: 100m
      memory: 50Mi
    requests:
      cpu: 20m
      memory: 5Mi
  terminationMessagePath: /dev/termination-log
  terminationMessagePolicy: File
  volumeMounts:
  - mountPath: /var/log/tidb
    name: slowlog
baseImage: <your_private_docker_hub_username>/tidb-ent:v5.4.0
plugins:
- audit-1
```

Use TiUP to deploy the plugin

Add the plugin to an existing cluster

1. Create a file on all target servers.

```
tiup cluster exec cluster_name --command "mkdir {{.DeployDir}}/plugin" -R tidb
```

2. Upload the plugin to all target servers.

```
tiup cluster push cluster_name ~/Download/audit-1.so {{.DeployDir}}/plugin/audit-1.so -R tidb
```

3. Change the TiDB configurations.

```
tiup cluster edit-config cluster_name
```

In the config field of each TiDB node, configure `plugin.dir` as the absolute path. `plugin.load` specifies the name of the plugin.

```
tidb_servers:
-  host: 172.16.7.213
   ssh_port: 22
   port: 4000
   status_port: 10080
   deploy_dir: /tidb-deploy/tidb-4000
   log_dir: /tidb-deploy/tidb-4000/log
   arch: amd64
   os: linux
   config:
      plugin.dir: /tidb-deploy/tidb-4000/plugin
      plugin.load: audit-1
```

4. Restart all TiDB nodes.

```
tiup cluster reload cluster_name -R tidb
```

Tips:

- In step 3, if all nodes have the same deployment path, you can directly configure it globally in server_configs.tidb to avoid manually modifying each node. If the paths are different, you can modify the path of the plugin in steps 1 and 2 to be the same path.
- If you do not need to deploy the plugin on all TiDB nodes, you can specify the node (the ID shown by the tiup cluster display) with the -N parameter.

Deploy the plugin during installation

Similar to steps in “Add the plugin to an existing cluster”. You need to configure `plugin.load` and `plugin.dir` before deployment, and run `tiup cluster exec` and `tiup cluster push` before starting the cluster.

Scale out

Similar to steps in “Add the plugin to an existing cluster”. You need to specify the node that is scaled out (the ID shown by the tiup cluster display) with the -N parameter.

Note that you need to upload the plugin file (audit-1.so in the above example) to the target server while the scaling is in progress, and make sure you finish uploading before the scaling ends. Otherwise an error will occur.

If an error occurs during scaling, you can perform the following to reinstall the plug-in.

1. Scale out the cluster again.
2. Upload the plug-in file after the scaling is completed.
3. Run the `tiup cluster start cluster_name command` to restart the cluster.

Upgrade the cluster

1. Delete the old plugin.

```
tiup cluster exec cluster_name --command "rm {{.DeployDir}}/plugin/audit-1.so" -R tidb
```

2. Upload the new plugin to all target servers. Use the same name to save you from changing the configurations.

```
tiup cluster push cluster_name ~/Download/audit-1_new.so {{.DeployDir}}/plugin/audit-1.so -R tidb
```

Note: The domain name cannot contain "-" (such as `tidb-server1`). Otherwise the command will fail to run.

3. Upgrade the cluster.

```
tiup cluster upgrade cluster_name version
```

Events of the plugin

TiDB audit plugin divides all the events into 3 categories:

CONNECTION: Connection events and authentication events.

Example of the CONNECTION logs:

- A successful authentication

  ```
  [2022/04/18 21:50:29.669 +08:00] [INFO] [logger.go:70] [ID=16502898290] [TIMESTAMP=2022/04/18 21:50:29.669 +08:00] [EVENT_CLASS=CONNECTION] [EVENT_SUBCLASS=Disconnect] [STATUS_CODE=0] [COST_TIME=0] [HOST=127.0.0.1] [CLIENT_IP=127.0.0.1] [USER=root] [DATABASES="[]"] [TABLES="[]"] [SQL_TEXT=] [ROWS=0] [CLIENT_PORT=39290] [CONNECTION_ID=11] [CONNECTION_TYPE=Socket] [SERVER_ID=1] [SERVER_PORT=4000] [DURATION=4522.55501] [SERVER_OS_LOGIN_USER=] [OS_VERSION="Centos 6"] [CLIENT_VERSION=] [SERVER_VERSION=v5.2.0] [AUDIT_VERSION=] [SSL_VERSION=v1.2.0] [PID=32638] [Reason=]
  ```

- A failed authentication

  ```
  [2022/04/18 21:50:40.198 +08:00] [INFO] [logger.go:70] [ID=16502898400] [TIMESTAMP=2022/04/18 21:50:40.198 +08:00] [EVENT_CLASS=CONNECTION] [EVENT_SUBCLASS=Reject] [STATUS_CODE=0] [COST_TIME=0] [HOST=127.0.0.1] [CLIENT_IP=127.0.0.1] [USER=ccd] [DATABASES="[]"] [TABLES="[]"] [SQL_TEXT=] [ROWS=0] [CLIENT_PORT=39296] [CONNECTION_ID=15] [CONNECTION_TYPE=Socket] [SERVER_ID=1] [SERVER_PORT=4000] [DURATION=0] [SERVER_OS_LOGIN_USER=] [OS_VERSION=”Centos 6"] [CLIENT_VERSION=] [SERVER_VERSION=v5.2.0] [AUDIT_VERSION=] [SSL_VERSION=v1.2.0] [PID=32638] [Reason="[server:1045]Access denied for user 'ccd'@'127.0.0.1' (using password: NO)"]
  ```

TABLE_ACCESS: SQL queries related to tables.

Example of the TABLE_ACCESS logs:

- Queries on a table(`select * from t`)

  ```
  [2022/04/18 22:00:47.085 +08:00] [INFO] [logger.go:70] [ID=16502904470] [TIMESTAMP=2022/04/18 22:00:47.085 +08:00] [EVENT_CLASS=TABLE_ACCESS] [EVENT_SUBCLASS=Select] [STATUS_CODE=0] [COST_TIME=0] [HOST=127.0.0.1] [CLIENT_IP=127.0.0.1] [USER=root] [DATABASES="[test]"] [TABLES="[t]"] [SQL_TEXT="select * from `t`"] [ROWS=0] [CONNECTION_ID=19] [CLIENT_PORT=39522] [PID=32638] [COMMAND=Query] [SQL_STATEMENTS=Select]
  ```

- Drop a table(`drop table t`)

  ```
  [2022/04/18 22:02:40.775 +08:00] [INFO] [logger.go:70] [ID=16502905600] [TIMESTAMP=2022/04/18 22:02:40.775 +08:00] [EVENT_CLASS=TABLE_ACCESS] [EVENT_SUBCLASS=DropTable] [STATUS_CODE=0] [COST_TIME=0] [HOST=127.0.0.1] [CLIENT_IP=127.0.0.1] [USER=root] [DATABASES="[test]"] [TABLES="[t,t1]"] [SQL_TEXT="drop table `t` , `t1`"] [ROWS=0] [CONNECTION_ID=19] [CLIENT_PORT=39522] [PID=32638] [COMMAND=Query] [SQL_STATEMENTS=DropTable]
  ```

Note:

- TABLE_ACCESS events are NOT logged by default. See "5. Modify audit configuration" on how to configure those events.
- If the statement fails, they are NOT logged.

GENERAL: SQL queries that don't access tables.

Example of the GENERAL logs:

- `show tables` statement:

  ```
  [2022/04/18 22:10:34.845 +08:00] [INFO] [logger.go:70] [ID=16502910340] [TIMESTAMP=2022/04/18 22:10:34.845 +08:00] [EVENT_CLASS=GENERAL] [EVENT_SUBCLASS=] [STATUS_CODE=0] [COST_TIME=0] [HOST=127.0.0.1] [CLIENT_IP=127.0.0.1] [USER=root] [DATABASES="[]"] [TABLES="[]"] [SQL_TEXT="show tables"] [ROWS=0] [CONNECTION_ID=19] [CLIENT_PORT=39522] [PID=32638] [COMMAND=Query] [SQL_STATEMENTS=Show]
  ```

- `begin` statement:

  ```
  [2022/04/18 22:06:53.198 +08:00] [INFO] [logger.go:70] [ID=16502908130] [TIMESTAMP=2022/04/18 22:06:53.198 +08:00] [EVENT_CLASS=GENERAL] [EVENT_SUBCLASS=] [STATUS_CODE=0]
  ```

```
[COST_TIME=0] [HOST=127.0.0.1] [CLIENT_IP=127.0.0.1] [USER=root] [DATABASES="[]"] [TABLES="[]"]
[SQL_TEXT=begin] [ROWS=0] [CONNECTION_ID=19] [CLIENT_PORT=39522] [PID=32638]
[COMMAND=Query] [SQL_STATEMENTS=Begin]
```

View plugin status

You can view the current plugin loading and enabling status by running the `show plugins` command in MySQL:

- Name: plugin name
- Status: plugin status, including the loading status (uninitialized, ready) and the enabling status (enable, disable)
- Type: plugin type
- Library: location of the plugin binary file
- Version: plugin version

```
mysql> show plugins;
+-------+--------------+-------+-----------------------------------------------------------------------------------+---------+---------+
| Name  | Status       | Type  | Library                                                                           | License | Version |
+-------+--------------+-------+-----------------------------------------------------------------------------------+---------+---------+
| audit | Ready-enable | Audit | /home/robi/code/go/src/github.com/pingcap/enterprise-plugin/audit/audit-1.so |         | 1       |
+-------+--------------+-------+-----------------------------------------------------------------------------------+---------+---------+
1 row in set (0.00 sec)
```

Enable or disable auditing

By default, the CONNECTION and GENERAL events are audited after enabling the plugin. You can enable or disable the plugin on any TiDB node, because the command takes effect on all TiDB nodes in the cluster once it is successfully executed on one TiDB node.

To enable the audit plugin, run the following command:

```
mysql> admin plugins enable audit;
```

If you do not need to audit the CONNECTION and GENERAL events, run the following command to disable the audit:

```
mysql> admin plugins disable audit;
```

```
mysql> admin plugins disable audit;
Query OK, 0 rows affected (0.03 sec)

mysql> show plugins;
+-------+--------------+-------+-----------------------------------------------------------------------------------+---------+---------+
| Name  | Status       | Type  | Library                                                                           | License | Version |
+-------+--------------+-------+-----------------------------------------------------------------------------------+---------+---------+
| audit | Ready-disable | Audit | /home/robi/code/go/src/github.com/pingcap/enterprise-plugin/audit/audit-1.so |         | 1       |
+-------+--------------+-------+-----------------------------------------------------------------------------------+---------+---------+
1 row in set (0.00 sec)

mysql> admin plugins enable audit;
Query OK, 0 rows affected (0.00 sec)

mysql> show plugins;
+-------+--------------+-------+-----------------------------------------------------------------------------------+---------+---------+
| Name  | Status       | Type  | Library                                                                           | License | Version |
+-------+--------------+-------+-----------------------------------------------------------------------------------+---------+---------+
| audit | Ready-enable | Audit | /home/robi/code/go/src/github.com/pingcap/enterprise-plugin/audit/audit-1.so |         | 1       |
+-------+--------------+-------+-----------------------------------------------------------------------------------+---------+---------+
1 row in set (0.00 sec)
```

Modify TABLE_ACCESS configurations

TABLE_ACCESS events are not audited by default. If you want to enable it, you need to modify the configuration.

You can implement fine-grained audit on TABLE_ACCESS events by combinations of the username, database, table name, and statement type. After modifying the configuration, you need to execute the `flush tidb plugins audit` command to validate the changes on all TiDB nodes.

Modify tidb_audit_table_access

When the audit plugin is started for the first time, a plugin configuration table named `tidb_audit_table_access` is created in TiDB's mysql library. You can modify the configuration table to control whether to audit the TABLE_ACCESS event.

```
mysql> show create table mysql.tidb_audit_table_access;
CREATE TABLE `tidb_audit_table_access` (
 `id` int(11) NOT NULL AUTO_INCREMENT,
 `user` varchar(32) NOT NULL COMMENT 'username need be audit(regex)',
```

```
 `db` varchar(32) NOT NULL COMMENT 'db name need be audit(regex)',
 `tbl` varchar(32) NOT NULL COMMENT 'table name need be audit(regex)',
 `access_type` varchar(128) NOT NULL COMMENT 'table access types split by comma',
 `create_time` timestamp NULL DEFAULT NULL COMMENT 'record create time',
 `update_time` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP COMMENT 'record last update time',
 PRIMARY KEY (`id`),
 UNIQUE KEY `user` (`user`,`db`,`tbl`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin
```

Parameters in the preceding commands are described as follows:

- user: the user who enables the TABLE_ACCESS event auditing
- db: the database that enables the TABLE_ACCESS event auditing
- tbl: the name of the table that enables the TABLE_ACCESS event auditing
- access_type: the type of the statement that enables the TABLE_ACCESS event auditing. The type is the same as the subClass mentioned above. It supports splitting by multiple commas, and `null` represents auditing all events.

Note: `user`, `db`, and `tbl` uses *regular expressions* and are case-sensitive.

Examples:

- TABLE_ACCESS enabled for all users, databases, and tables:

  ```
  mysql> insert into mysql.tidb_audit_table_access (user, db, tbl, access_type) values
  ('.*', '.*', '.*', '');
  ```

- Audit any SQL query that any user uses to access the `account` table in the `ac` database:

  ```
  mysql> insert into mysql.tidb_audit_table_access (user, db, tbl, access_type) values
  ('.*', '^ac$', '^account$', '');
  ```

- Audit any SQL query that users named starts with `op_` to access the `account` all tables from all databases:

  ```
  mysql> insert into mysql.tidb_audit_table_access (user, db, tbl, access_type) values ('op_.*', '.*', '.*', '');
  ```

  Note that the match rules are NOT wildcard-style. For example, if `user` is set to `abc*`, it matches "ab", "abc", "abcc", "abccc", and so on.

The following are supported access_types. You can separate them by multiple commas. An empty string means all are audited.

- Insert
- Select
- Update
- Replace
- Delete
- LoadData
- Alter
- DropTable
- CreateTable

Refresh configuration

After modifying the `mysql.tidb_audit_table_access` table, you need to refresh it by running the following command. This notifies all TiDB nodes in the cluster to load the latest configuration from the table. If the `flush` fails, you need to retry the operation.

```
mysql> flush tidb plugins audit;
Query OK, 0 rows affected (0.00 sec)
```

Output audit logs

After the plugin is loaded, different types of audit logs are output to the log file called `tidb-audit.log` in the same directory as the TiDB configuration item `slow-query-file`.

Currently, the following types of audit events are supported.

Event Class	SubEvent	Enable/Disable	Description
CONNECTION	Connected Disconnect ChangeUser PreAuth Reject	Enabled by default; To cancel auditing these events, disable the plugin.	Connection events and authentication events
TABLE_ACCESS	Insert Select Update Replace Delete LoadData Alter DropTable CreateTable	Disabled by default; To audit these events, specify “db - user - table - event” in the configuration table.	SQL queries related to tables; Parameters in SQL statements are replaced with “?” for data masking.
GENERAL	-	Enabled by default; To cancel auditing this event, disable the plugin.	Other SQL queries that don’t access tables.

Log output

The log output of all types of events begins with the same set of fields. Following the first three fields output by the log library are event fields as follows:

- ID: event ID
- TIMESTAMP: event time
- EVENT_CLASS: event type
- EVENT_SUBCLASS: event subtype
- STATUS_CODE: response status of the statement
- COST_TIME: time consumed by the statement
- HOST: server IP
- CLIENT_IP: client IP
- USER: login username
- DATABASE: event-related database
- TABLES: event-related table name
- SQL_TEXT: masked SQL statement
- ROWS: the number of rows affected (“0” indicates that no rows are affected)

There might be redundant data, or output even when there is no data. This is to facilitate the current log analysis system to process data.

CONNECTION

Connection events also contain the following information in the output:

- CLIENT_PORT: client port number
- CONNECTION_ID: connection ID
- CONNECTION_TYPE: connect via `socket` or `unix-socket`
- SERVER_ID: TiDB server ID
- SERVER_PORT: the port that the TiDB server uses to listen to the MySQL protocol
- SERVER_OS_LOGIN_USER: the username of the TiDB process startup system
- OS_VERSION: the version of the operating system where the TiDB server is located
- SSL_VERSION: the current SSL version of TiDB

- PID: the PID of the TiDB process

Example:

```
[2019/07/24 22:13:30.099 +08:00] [INFO] [logger.go:71] [ID="15640224010"] [TIMESTAMP=2019/07/24 22:13:30.099 +08:00] [EVENT_CLASS=CONNECTION] [EVENT_SUBCLASS=Disconnect] [STATUS_CODE=0] [COST_TIME=0] [HOST=127.0.0.1] [CLIENT_IP=127.0.0.1] [USER=root] [DATABASES="[test]"] [TABLES="[]"] [SQL_TEXT=] [ROWS=0] [CLIENT_PORT=45814] [CONNECTION_ID=1] [CONNECTION_TYPE=Socket] [SERVER_ID=1] [SERVER_PORT=4000] [DURATION=1233372.167907] [SERVER_OS_LOGIN_USER=robi] [OS_VERSION="Linux 5.0.0-20-generic.x86_64"] [CLIENT_VERSION=] [SERVER_VERSION=v2.1.15-16-g136e334df-dirty] [AUDIT_VERSION=] [SSL_VERSION=v1.2.0] [PID=24108]
```

TABLE_ACCESS

TABLE_ACCESS events also contain the following information in the output:

- CONNECTION_ID: connection ID
- COMMAND: the command type of the MySQL protocol
- SQL_STATEMENTS: the SQL statement type
- PID: the PID of the TiDB process

Example:

```
[2019/07/24 22:35:01.640 +08:00] [INFO] [logger.go:71] [ID="15640226010"] [TIMESTAMP=2019/07/24 22:35:01.640 +08:00] [EVENT_CLASS=TABLE_ACCESS] [EVENT_SUBCLASS=Insert] [STATUS_CODE=0] [COST_TIME=31581.623] [HOST=127.0.0.1] [CLIENT_IP=127.0.0.1] [USER=root] [DATABASES="[mysql]"] [TABLES="[t1]"] [SQL_TEXT="insert into t1 ( a ) values ( ? )"] [ROWS=1] [CONNECTION_ID=2] [CLIENT_PORT=47430] [PID=24108] [COMMAND=Query] [SQL_STATEMENTS=Insert]
```

GENERAL

GENERAL events also contain the following information in the output:

- CONNECTION_ID: connection ID
- COMMAND: the command type of the MySQL protocol
- SQL_STATEMENTS: the SQL statement type
- PID: the PID of the TiDB process

Example:

```
[2019/07/24 22:31:22.612 +08:00] [INFO] [logger.go:71] [ID="15640334010"] [TIMESTAMP=2019/07/24 22:31:22.612 +08:00] [EVENT_CLASS=GENERAL] [EVENT_SUBCLASS=] [STATUS_CODE=0] [COST_TIME=67.6] [HOST=127.0.0.1] [CLIENT_IP=127.0.0.1] [USER=root] [DATABASES="[]"] [TABLES="[]"] [SQL_TEXT="set @@tidb_max_chunk_size = ?"] [ROWS=0] [CONNECTION_ID=2] [CLIENT_PORT=47430] [PID=24108] [COMMAND=Query] [SQL_STATEMENTS=Set]
```

Clear the audit logs

TiUP does not automatically clean up the audit logs generated by the audit plugin. If you want to clear the audit logs, run the following command:

```
tiup cluster clean <cluster-name> --audit-log
```

Note:

- This command deletes all audit logs by default.
- The cleanup operation is irreversible. Back up your data before using it.