

# Changefeed User doc (With CLI)

## Prerequisite

Set up [tidbcloud cli](#) and upgrade at least to 1.0.0-beta.10

## Changefeed CRUD

### Create a changefeed

General:

Code block

```
1  # to kafka
2  ticloud serverless changefeed create -c <cluster-id> --name kakfa-cf --type
   KAFKA --kafka <kafka-json> --filter <filter-json>
3
4  # to mysql
5  ticloud serverless changefeed create -c <cluster-id> --name mysql-cf --type
   MYSQL --mysql <mysql-json> --filter <filter-json>
```

Details:

- [Create a changefeed to AliCloud RDS with private link](#)
- [Create a changefeed to kafka with public network](#)

### Pause a changefeed

Code block

```
1  ticloud serverless changefeed pause -c <cluster-id> --changefeed-id
   <changefeed-id>
```

### Resume a changefeed

Code block

```
1  ticloud serverless changefeed resume -c <cluster-id> --changefeed-id
   <changefeed-id>
```

## Edit a changefeed

Code block

```
1  ticloud serverless changefeed edit -c <cluster-id> --changefeed-id <changefeed-id> --name <new-name> --kafka <full-specified-kafka> --filter <full-specified-filter>
2
3  ticloud serverless changefeed edit -c <cluster-id> --changefeed-id <changefeed-id> --name <new-name> --mysql <full-specified-mysql> --filter <full-specified-filter>
```

## Delete a changefeed

Code block

```
1  ticloud serverless changefeed delete -c <cluster-id> --changefeed-id <changefeed-id>
```

## Changefeed to MySQL

### Private Link (AliCloud RDS)

#### before you start

1. set up the endpoint service. see [set up endpoint service](#)

#### Create a private link connection

1. Create a private link connection
  - a. # You can get the endpoint-service-name from Alibaba Cloud console, where you created the endpoint service

Code block

```
1  ticloud s plc create -c <cluster-id> --display-name <dispaly-name> --type ALICLOUD_ENDPOINT_SERVICE --alicloud.endpoint-service-name <name>
```

2. Get the state of the private link connection

- a. You can get the private-link-connection-id from the previous step.

```

1  # Code block
2
3  # Example:
4  {
5    "alicloudEndpointService": {
6      "name": "com.aliyuncs.privatelink.ap-southeast-1.epsrv-
7      t4nxgk5mx4uhowcftlqc"
8    },
9    "clusterId": "10225155406620916674",
10   "createTime": "2025-11-28T05:24:43Z",
11   "createdBy": "apikey-MCTGR3Jv",
12   "displayName": "ali-rds-dev",
13   "message": "",
14   "privateLinkConnectionId": "plc-uwq4hgsn5fa4jg2xchfij5bau4",
15   "state": "PENDING_ACCEPTANCE",
16   "type": "ALICLOUD_ENDPOINT_SERVICE",
17   "updateTime": "2025-11-28T05:25:51Z"
18 }

```

3. Allow the `endpoint connections` on the Alibaba Cloud console when the private link connection state is in `PENDING_ACCEPTANCE`.

VPC / Endpoint Service / epsrv-t4nxgk5mx4uhowcftlqc

← **com.aliyuncs.privatelink.ap-southeast-1.epsrv-t4nxgk5mx4uhowcftlqc** Delete ↻

Basic Information **Endpoint Connections** Service Whitelist Monitor

Endpoint ID Q Enter ↻

Endpoint ID	Monitor	Endpoint VPC	Endpoint Owner	Connection Modification Time	Status <span>⌵</span>	Bandwidth Limit	Actions
+ <a href="#">ep-t4ni347a04a5eab468e8</a>		<a href="#">vpc-t4n86e3djfwnt5bbtaxq4</a>	Current Account	Nov 28, 2025, 13:24:56	<span>Disconnected</span>	5120 Mbps	<a href="#">Allow</a>

4. Wait private link connection's state changes to active (around half a minute)

Code block

```

1  ticloud s plc get -c <cluster-id> --private-link-connection-id <private-link-
2  connection-id>
3
4  {
5    "alicloudEndpointService": {
6      "name": "com.aliyuncs.privatelink.ap-southeast-1.epsrv-
7      t4nxgk5mx4uhowcftlqc"
8    },
9    "clusterId": "10225155406620916674",
10   "createTime": "2025-11-28T05:24:43Z",
11   "createdBy": "apikey-MCTGR3Jv",

```

```
10     "displayName": "ali-rds-dev",
11     "message": "",
12     "privateLinkConnectionId": "plc-uwq4hgsn5fa4jg2xchfij5bau4",
13     "state": "ACTIVE",
14     "type": "ALICLOUD_ENDPOINT_SERVICE",
15     "updateTime": "2025-11-28T05:27:22Z"
16 }
```

## Create a changefeed based on the private link connection

Please ensure the table you want to replicate exists in your RDS.

### 1. Create the changefeed based on the private link connection

- change `filter`, `mysql` as you need.

Code block

```
1  ticloud s changefeed create --type "MYSQL" --filter '{"filterRule":
  ["db.table"],"mode":"FORCE_SYNC"}' --mysql '{"network":
  {"networkType":"PRIVATE_LINK","privateLink":{"privateLinkConnectionId": "plc-
  uwq4hgsn5fa4jg2xchfij5bau4","port":3306}},"authentication":
  {"userName":"root","password":"xxx","enableTls":false}}' -c
  10225155406620916674 --display-name 'ali-rds-plc'
```

### 2. Waiting for the changefeed to be running

- You can get the changefeed id from the previous step.

Code block

```
1  ticloud s changefeed get -c <cluster-id> -f <changefeed-id>
```

## Changefeed to Kafka

### Public Network Access

Code block

```
1  ticloud serverless changefeed create -c <cluster-id> --name kafka-cf --type
  KAFKA --kafka <kafka-json> --filter <filter-json>
```

1. See [filter configuration](#) to get more information about --filter
2. See [Kafka configuration](#) to get more information about --kafka

A demo to sync `db.table` to kafka `test` topic in DEBEZIUM format

Code block

```
1  ticloud s changefeed create \  
2    -c <cluster-id> \  
3    --display-name test \  
4    --type KAFKA \  
5    --filter '{  
6      "filterRule": ["db.table"],  
7      "mode": "FORCE_SYNC"  
8    }' \  
9    --kafka '{  
10     "network": {  
11       "networkType": "PUBLIC",  
12       "publicEndpoints": "<host:port>"  
13     },  
14     "broker": {  
15       "kafkaVersion": "VERSION_3XX",  
16       "compression": "NONE"  
17     },  
18     "authentication": {  
19       "authType": "<authType>",  
20       "userName": "<username>",  
21       "password": "<password>",  
22       "enableTls": [true/false]  
23     },  
24     "dataFormat": {  
25       "protocol": "DEBEZIUM"  
26     },  
27     "topicPartitionConfig": {  
28       "dispatchType": "ONE_TOPIC",  
29       "defaultTopic": "<topic name>",  
30       "replicationFactor": 3,  
31       "partitionNum": <your partitionNum>,  
32       "partitionDispatchers": [{"partitionType": "TS"}]  
33     }  
34   }'
```

Please fill in :

- <cluster-id>
- "publicEndpoints":"<host:port>"

- filterRule: input your desired db and table
- Kafka authentication into : <authType>,<username>,<password>,enableTls

- eg:

```
{ "authType": "SASL_SCRAM_SHA_512", "userName": "cdc", "password": "xxxxxx", "enableTls": true }
```

- if you don't have any authentication , just use "authentication":

```
{ "authType": "DISABLE", "enableTls": false }
```

- Topic name
- More configuration please see [kafka configuration](#)

## Ref

### Deploy ticloud CLI

- Install via script (mac/linux)

Code block

```
1 curl https://raw.githubusercontent.com/tidbcloud/tidbcloud-cli/main/install.sh
| sh
```

- authentication:

Code block

```
1 ticloud auth login // OAuth
2
3 or
4
5 ticloud config create // OPEN API
```

- Test CLI

Code block

```
1 ticloud serverless list
2 Choose the project:
3 > [x] likun(1372813089196101288)
4
5 ID                      DisplayName  State  Version  CloudProvider  Region
   CreateTime
```

6 10693632960012990795 Cluster-EU ACTIVE v7.5.6 aws Frankfurt  
(eu-central-1) 2024-09-16T09:28:54Z

More info : <https://docs.pingcap.com/tidbcloud/get-started-with-cli/?plan=starter>

## Set up endpoint service

Refer to the official doc to learn how to create an endpoint service

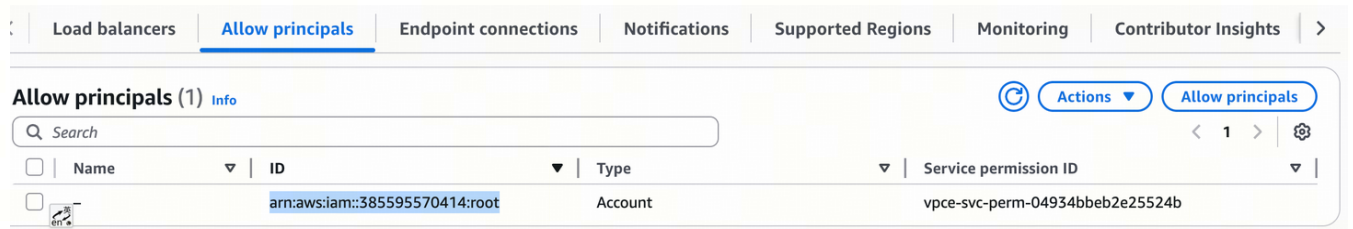
- AWS: <https://docs.aws.amazon.com/vpc/latest/privatelink/create-endpoint-service.html>
- Alicloud: <https://www.alibabacloud.com/help/en/privatelink/user-guide/create-and-manage-endpoint-services>

During the creation, you will need the TiDB Cloud Account and azs

Code block

```
1 ticloud s plc zones -c 10031506923317518814
2
3 {
4   "accountId": "385595570414",
5   "azIds": [
6     "use1-az2",
7     "use1-az4",
8     "use1-az1"
9   ]
10 }
```

- TiDB Cloud Account: You need to allow TiDB Cloud Account to visit your endpoint service
  - AWS: `arn:aws:iam::<account>:root`



- Alicloud: `acs:ram:*:<account>:*`

← com.aliyuncs.privatelink.ap-southeast-1.epsrv-t4ncyhflv6miol7ofko Delete ↻

Basic Information Endpoint Connections **Service Whitelist** Monitor

Add to Whitelist Whitelist Account ▼ Q Search by UID ↻

Whitelist Account	Actions
acs:ram*:5654539846535456:*	<span>Delete</span>

Items per Page 20 ▼ Total: 1 < 1/1 >

- TiDB Cloud Azs: your endpoint service needs to cover at least one of the TiDB Cloud azs

## Filter Configuration

Code block

```

1  {
2      "filterRule": ["test.t1", "test.t2"],
3      // "mode": "IGNORE_NOT_SUPPORT_TABLE", "FORCE_SYNC"
4      "mode": "IGNORE_NOT_SUPPORT_TABLE",
5      // use it when you want to filter the specified event
6      "eventFilterRule": [
7          {
8              "matcher": ["test.t1", "test.t2"],
9              "ignoreEvent": ["all dml", "all ddl"]
10         }
11     ]
12 }
```

- FilterRule: see <https://docs.pingcap.com/tidb/stable/table-filter/#syntax> for details
- Mode
  - IGNORE\_NOT\_SUPPORT\_TABLE: ignore the table that does not have a primary key or unique key
  - FORCE\_SYNC: sync all.
- ignoreEvents valid value

Code block

```

1  var ignoreEvents = []string{
2      "all dml",
3      "all ddl",
4      "insert",
5      "update",
6      "delete",
7      "create schema",
8  }
```



```
8      "drop schema",
9      "create table",
10     "drop table",
11     "rename table",
12     "truncate table",
13     "alter table",
14     "add table partition",
15     "drop table partition",
16     "truncate table partition",
17     "create view",
18     "drop view",
19     "modify schema charset and collate",
20     "recover table",
21     "rebase auto id",
22     "modify table comment",
23     "modify table charset and collate",
24     "exchange table partition",
25     "reorganize table partition",
26     "alter table partitioning",
27     "remove table partitioning",
28     "add column",
29     "drop column",
30     "modify column",
31     "set default value",
32     "add primary key",
33     "drop primary key",
34     "rename index",
35     "alter index visibility",
36     "alter ttl info",
37     "alter ttl remove",
38     "multi schema change",
39 }
```

## Example

Code block

```
1  --filter '{"filterRule":
    ["test.*"], "mode": "IGNORE_NOT_SUPPORT_TABLE", "eventFilterRule": [{"matcher":
    ["test.t1"], "ignoreEvent": ["all ddl"]}]]'
```

1. Filter the data in the test DB
2. Ignore the non-support table, for example: the table without pk
3. Do not need the ddl message in test.t1

# Kafka Configuration

Code block

```
1  {
2      "network": {
3          "networkType": "PUBLIC",
4          "publicEndpoints": "broker1:9092,broker2:9092"
5      },
6      "broker": {
7          "kafkaVersion": "VERSION_2XX",
8          "compression": "NONE"
9      },
10     "authentication": {
11         "authType": "DISABLE",
12         "userName": "",
13         "password": "",
14         "enableTls": false
15     },
16     "dataFormat": {
17         "protocol": "CANAL_JSON",
18         "enableTidbExtension": false,
19         "debeziumConfig": {
20             "disableOutputOldValue": false,
21             "disableSchema": false
22         },
23         "avroConfig": {
24             "decimalHandlingMode": "PRECISE",
25             "bigintUnsignedHandlingMode": "LONG",
26             "confluentSchemaRegistry": {
27                 "schemaRegistryEndpoints": "",
28                 "enableHttpAuth": false,
29                 "userName": "",
30                 "password": ""
31             },
32             "awsGlueSchemaRegistry": {
33                 "region": "",
34                 "name": "",
35                 "accessKeyId": "",
36                 "secretAccessKey": ""
37             }
38         }
39     },
40     "topicPartitionConfig": {
41         "dispatchType": "ONE_TOPIC",
42         "defaultTopic": "test-topic",
43         "topicPrefix": "_prefix",
```

```

44     "separator": "_",
45     "topicSuffix": "_suffix",
46     "replicationFactor": 1,
47     "partitionNum": 1,
48     "partitionDispatchers": [{
49         "partitionType": "TABLE",
50         "matcher": ["*.*"],
51         "indexName": "index1",
52         "columns": ["col1", "col2"]
53     }]
54 },
55     "columnSelectors": [{
56         "matcher": ["*.*"],
57         "columns": ["col1", "col2"]
58     }],
59     "outputRawChangeEvent": false,
60 }

```

- Network: only public is supported now.
- Broker: the kafka information
  - kafkaVersion: "VERSION\_2XX", "VERSION\_3XX"
  - Compression: "NONE", "GZIP", "LZ4", "ZSTD", "SNAPPY"
- Authentication:
  - authType: "DISABLE", "SASL\_PLAIN", "SASL\_SCRAM\_SHA\_256", "SASL\_SCRAM\_SHA\_512"
  - userName and password are required for SASL\_PLAIN, SASL\_SCRAM\_SHA\_256, or SASL\_SCRAM\_SHA\_512
- dataFormat
  - Protocol: "CANAL\_JSON", "AVRO", "OPEN\_PROTOCOL", "DEBEZIUM"
  - enableTidbExtension: `false` by default. When the output protocol is `canal-json`, if the value is `true`, TiCDC sends Resolved events and adds the TiDB extension field to the Kafka message. From v6.1.0, this parameter is also applicable to the `avro` protocol. If the value is `true`, TiCDC adds three TiDB extension fields to the Kafka message.
  - avroConfig: available when protocol is AVRO
  - debeziumConfig: available when protocol is DEBEZIUM
- topicPartitionConfig
  - dispatchType:
    - "ONE\_TOPIC": all messages route to one topic

- "BY\_TABLE": route by table name
  - "BY\_DATABASE": route by db name
- defaultTopic: the basic topic name
- topicPrefix, separator and topicSuffix: how to build the topic name when dispatchType is BY\_TABLE or BY\_DATABASE
- replicationFactor: replication number
- partitionNum:
- partitionDispatchers: how to route partition
  - partitionType:
    - "TABLE": by table name
    - "INDEX\_VALUE": by specified index value
    - "TS": by timestamp
    - "COLUMN": u specified a column
- columnSelectors: define it when you only need specified columns
- outputRawChangeEvent: <https://docs.pingcap.com/zh/tidb/stable/ticdc-changefeed-config/#output-raw-change-event>

## Example

Code block

```

1  --kafka '{
2    "network": {
3      "networkType": "PUBLIC",
4      "publicEndpoints": "broker1:9092,broker2:9092"
5    },
6    "broker": {
7      "kafkaVersion": "VERSION_3XX",
8      "compression": "NONE"
9    },
10   "authentication": {
11     "authType": "SASL_SCRAM_SHA_512",
12     "userName": "username",
13     "password": "xxx",
14     "enableTls": false
15   },
16   "dataFormat": {
17     "protocol": "CANAL_JSON",

```

```

18     "enableTidbExtension": false
19 },
20 "topicPartitionConfig": {
21     "dispatchType": "ONE_TOPIC",
22     "defaultTopic": "use1-cdc",
23     "replicationFactor": 2,
24     "partitionNum": 2,
25     "partitionDispatchers": [
26         {
27             "partitionType": "TABLE",
28             "matcher": ["*.*"]
29         }
30     ]
31 }
32 }'

```

1. Connect to Kafka with public endpoint, auth with SASL\_SCRAM\_SHA\_512
2. All messages will not be compressed
3. Use CANAL\_JSON format
4. All messages are routed to use1-cdc topic, partition based on table name
5. 2 partitions and 2 replicas

## How to dispatch different tables to different topics

It is recommended that the total number of topics be less than 1000.

1. Filter the DB and table you need

Code block

```

1  {
2      "filterRule": ["test.*", "test1.t"],
3  }

```

2. Define topic dispatch rule

Code block

```

1  "topicPartitionConfig": {
2      "dispatchType": "BY_TABLE",
3      "defaultTopic": "default-topic",

```

```
4      "topicPrefix": "prefix_",
5      "separator": "_",
6      "topicSuffix": "_suffix",
7      "replicationFactor": 1,
8      "partitionNum": 1,
9      "partitionDispatchers": [{
10         "partitionType": "TABLE",
11         "matcher": ["*.*"],
12     }]
13 },
```

Pay attention to `dispatchType`, `topicPrefix`, `separator` and `topicSuffix`

The format of the Topic expression is `[topicPrefix]{schema}[separator]{table}[topicSuffix]`.

- `topicPrefix`: Indicates the prefix of the Topic Name.
- `{schema}`: the db name
- `separator`: Indicates the delimiter between db name and table name.
- `{table}`: the table name
- `topicSuffix`: Indicates the suffix of the Topic Name.

In this example:

Table `test.t` will be send to topic `prefix_test_t_suffix`