CVP レポート：重複スケジュールのトラブルシューティング

目次

概要

本書では、Cisco Customer Voice Portal（CVP）レポート機能の重複スケジュールのトラブルシューティング プロセスについて説明します。

著者：Cisco TAC エンジニア、Aleksey Yankovskyy、Alexander Levichev

前提条件

要件

次の項目に関する知識が推奨されます。

- Microsoft Windows Server
- Cisco CVP
- Informix DB アクセス ツール

使用するコンポーネント

このドキュメントの情報は CVP サーバ バージョン 11.0 に基づいていますが、以前のバージョンにも適用できます。

本書の情報は、特定のラボ環境にあるデバイスに基づいて作成されたものです。 このドキュメントで使用するすべてのデバイスは、初期（デフォルト）設定の状態から起動しています。 稼働中のネットワークで作業を行う場合、コマンドの影響について十分に理解したうえで作業してください。

CVP レポートの重複したスケジュールのトラブルシュート

ciscoadmin データベースでは、**agg_schedule** テーブルがそれぞれの集約が実行される頻度を制御します。 アップグレード後には、古いスケジュールが存在しても、新しいスケジュールでテーブルがリロードされるケースもあります。 これにより各集約が 2 回実行される結果となります。 これはそれぞれの集約が 2 倍の行数と総数を挿入することになり、サマリ テーブルの精度に不利に影響します。

診断

agg_schedule テーブルに重複があるかどうか検証します。

cvp_dbadmin ユーザで CVP レポート サーバにログインします。

Windows [CMD] ツールを起動します。 **dbaccess** コマンドを入力します。 [Connection] タブ > [Connect] を選択します。 **cvp** データベース サーバを選択し、Enter を押します。 クレデンシャルを入力するように求められます。 **cvp_dbadmin** アカウントを使用します。

ciscoadmin@cvp データベースを選択します。 [Query-language] > [New] を選択します。 次のコマンドを実行します。

```
UNLOAD TO schedule.txt SELECT * FROM agg_schedule
```

schedule.txt ファイルを開きます。 CVP バージョンによっては、C:\Users\Administrator または C:\db\Informix\etc\sysadmin フォルダのいずれかにある場合があります。

これは固有の CVP データのスケジュール エントリである必要があります。 たとえばこの図で示すような、**call_15** または **applicationsummary_daily** です。

```
1  cvp_data|call_15||15|2016-09-07 07:33:57|dbdatetime|N|1|60 units day|
2  cvp_data|call_30|call_15|30|2000-01-01 00:00:00|dbdatetime|Y|2|60 units day|
3  cvp_data|call_hourly|call_15|60|2000-01-01 00:00:00|dbdatetime|Y|3|60 units day|
4  cvp_data|call_daily|call_15|DD|2016-09-07 07:33:57|dbdatetime|N|4|550 units day|
5  cvp_data|call_weekly|call_daily|WW|2016-09-07 07:33:57|dbdatetime|N|5|10 units year|
6  cvp_data|applicationsummary_15||15|2016-09-07 07:33:57|a.dbdatetime|N|6|60 units day|
7  cvp_data|applicationsummary_daily|applicationsummary_15|DD|2016-09-07 07:33:57|dbdatetime|N|7|550 units day|
8  cvp_data|applicationsummary_weekly|applicationsummary_daily|WW|2016-09-07 07:33:57|dbdatetime|N|8|10 units year|
9  cvp_data|call_monthly|call_daily|MM|2016-09-07 07:33:57|dbdatetime|N|9|40 units year|
10 cvp_data|applicationsummary_monthly|applicationsummary_daily|MM|2016-09-07 07:33:57|dbdatetime|N|10|40 units year|
11
```

重複したタイプが存在しないことを確認してください。 重複が見つかった場合は、回避策を実施します。

回避策

重複したスケジュールの削除

Windows Task Scheduler でサマリ ジョブ ([CVPSummary]) を無効にします。

schedule.txt ファイルを開き、重複するすべての行を削除します。 最初の 10 エントリのみが残る必要があります。

前述したように **ciscoadmin** データベースに接続し、このクエリを実行します。 このコマンドは、**agg_schedule** テーブルからのすべてのエントリを削除します。

- ```
  DELETE FROM agg_schedule WHERE 1=1;
  ```

新しい値を schedule.txt ファイルから口ードします。このファイルには agg_schedule テーブルへの重複は含まれていません。

- ```
  LOAD FROM schedule.txt INSERT INTO agg_schedule;
  ```

agg_schedule テーブルには重複が存在しないことを確認します。 Schedule1.txt ファイルの出力には、10 個のエントリのみが含まれている必要があります。

- ```
  UNLOAD TO schedule1.txt SELECT * FROM agg_schedule;
  ```

重複したエントリのテーブルからの削除

15 分のテーブルは、他のすべてのテーブルが入力されるベースであるため、最初に修復する必要があります。

cvp_data データベースに接続します。

call_15 テーブルにこれらのコマンドを実行します。

```
SELECT distinct * FROM call_15 into temp t1 with no log;
TRUNCATE table call_15;
INSERT into call_15 select * from t1;
DROP table t1;
```

applicationsummary_15 テーブルに同じ手順を繰り返します。

```
SELECT distinct * from applicationsummary_15 into temp t1 with no log;
TRUNCATE table applicationsummary_15;
INSERT into applicationsummary_15 select * from t1;
DROP table t1;
```

> **注: 60 日より前に問題が始まっている場合は、上述の手順を日次、週次、および月次テーブルに対して繰り返します。**

最近の実行のリセット

15 分のテーブルの [lastrun] フィールドをリセットします。

15 分のテーブルの更新時間を見つけます。 cvp_data データベースに対してこれらのコマンドを実行します。

```
SELECT max(dbdatetime) FROM applicationsummary_15;
SELECT max(dbdatetime) FROM call_15;
```

最後の更新以降の日数を書き留めます。

このクエリを実行して、ciscoadmin DB で 15 分のテーブルの最近の実行をリセットします。 この例では、15 分のテーブルは最後は 17 日前に更新されました。
```

```
UPDATE ciscoadmin:agg_schedule SET lastrun = (current year to day - 17 units day) WHERE
dst_tabname LIKE 'call_15';
UPDATE ciscoadmin:agg_schedule SET lastrun = (current year to day - 17 units day) WHERE
dst_tabname LIKE 'applicationsummary_15';
```

注: コマンドから "17" を、これらの 2 つのテーブルの各手順から得た日数に置き換えます。

この後、手順 15 分のテーブルは修正されます。

15 分のテーブルは直近 60 日のデータを保持するため、日次、週次、および月次テーブルの lastrun 値をリセットし、これらのテーブルに対してすべてのデータを 60 日前まで削除します。 この方法は、aggregation.bat のプロセスが開始される次回に、日次、週次、月次テーブルに正しい値でデータが確実に入力されるようにします。

日次、週次、および月次テーブルについて、ciscoadmin データベースに対して実行されたこれらのコマンドで lastrun をリセットします。

```
UPDATE ciscoadmin:agg_schedule SET lastrun = (current year to day - 60 units day) WHERE
dst_tabname LIKE 'call_daily';
UPDATE ciscoadmin:agg_schedule SET lastrun = (current year to day - 60 units day) WHERE
dst_tabname LIKE 'call_weekly';
UPDATE ciscoadmin:agg_schedule SET lastrun = (current year to day - 60 units day) WHERE
dst_tabname LIKE 'call_monthly;
UPDATE ciscoadmin:agg_schedule SET lastrun = (current year to day - 60 units day) WHERE
dst_tabname LIKE 'applicationsummary_daily;
UPDATE ciscoadmin:agg_schedule SET lastrun = (current year to day - 60 units day) WHERE
dst_tabname LIKE 'applicationsummary_weekly;
UPDATE ciscoadmin:agg_schedule SET lastrun = (current year to day - 60 units day) WHERE
dst_tabname LIKE 'applicationsummary_monthly;
```

日次、週次、および月次テーブルのすべてのデータを、60 日前まで削除します。

```
DELETE FROM cvp_data:call_daily WHERE dbdatetime > (current - 60 units day);
DELETE FROM cvp_data:call_weekly WHERE dbdatetime > (current - 60 units day);
DELETE FROM cvp_data:call_monthly WHERE dbdatetime > (current - 60 units day);

DELETE FROM cvp_data&colon;applicationsummary_daily WHERE dbdatetime > (current - 60 units day);
DELETE FROM cvp_data&colon;applicationsummary_weekly WHERE dbdatetime > (current - 60 units
day);
DELETE FROM cvp_data&colon;applicationsummary_monthly WHERE dbdatetime > (current - 60 units
day);
```

集約プロセスのタスク スケジューラのサマリ ジョブ CVPSummary を有効にして、再開します。

確認

このセクションでは、設定が正常に機能していることを確認します。

サマリ テーブルが、cvp_db データベースに対して実行されたこれらのコマンドを使用して確実にアップデートされるようにします。

```
SELECT MAX(dbdatetime) FROM applicationsummary_15;
SELECT max(dbdatetime) FROM applicationsummary_daily;
SELECT max(dbdatetime) FROM applicationsummary_weekly;
```

```
SELECT max(dbdatetime) FROM applicationsummary_monthly;

SELECT MAX(dbdatetime) FROM call_15;
SELECT MAX(dbdatetime) FROM call_daily;
SELECT MAX(dbdatetime) FROM call_weekly;
SELECT MAX(dbdatetime) FROM call_monthly;
```

関連情報

- CVP レポートの設定例を使用した Squirrel SQL クライアントの統合
- テクニカル サポートとドキュメント