Informix の高 CPU 使用率

内容

概要

このドキュメントでは、ローカル UCCX データベース アクセスを必要とする Unified Contact Center Express（UCCX）アクティビティの速度が低下する仕組みについて説明します。これが原因で、AppAdmin ページの読み込みが遅くなり、AppAdmin の更新が有効になるまでに時間がかかり、ウォールボード クエリに対する応答が遅延し、Workforce Manager が UCCX データをクエリできなくなるなど、その他にもパフォーマンスと安定性の問題が発生します。

CLIで**show process loadコマンド**を入力すると、uccxoninitが大量のCPUを消費することがわかります。uccxoninitプロ**セス**は、UCCXサーバで実行するUCCX Informixデータベースインスタンスを表します。

著者：Cisco TACエンジニアSridhar Chandrasekharan、Ryan LaFountain、Ben Wollak

機能情報

UCCXアプリケーションをサポートするデータベースエンジンは、IBMのInformixです。UCCXのAppAdminページに追加され、UCCXアプリケーションによって生成された構成および履歴情報は、UCCX Informixインスタンスに保存されます。

UCCXアプリケーションには、ウォールボードアプリケーション、品質管理、ワークフォース管理、カスタム履歴レポートの目的で情報を抽出するためにUCCXデータベースに直接アクセスできる3つのユーザが用意されています。

次に、ユーザ情報、各ユーザの権限、および各ユーザの意図する目的について説明します。

- **uccxhruser**：このユーザは、UCCXデータベース内の多数の構成テーブルおよび履歴テーブルに対する選択権限を持ち、カスタム履歴レポートおよびCisco Unified Workforce Management(WFM)にのみ使用する必要があります。 このユーザーが実行するクエリとストアドプロシージャは、複雑で長い間実行されるクエリを実行する可能性があります。一般的な履歴レポートまたはWFMユーザのプロファイルにより、これらのクエリおよびストアドプロシージャは、ウォールボードアプリケーションで発生する頻度ほど頻繁に実行しないでください。

  多くのウォールボードアプリケーションでは、uccxhruserがアクセスできる構成テーブルと履歴テーブルに含まれるデータが必要です。ウォールボードアプリケーションの目的で、このユーザを使用してUCCXデータベースに対して複雑で頻繁にクエリをすることはサポート

されています。

- **ucxworkforce** - uccxworkforceユーザは、チーム、リソース、およびスーパーバイザのテーブルにアクセスでき、Cisco Unified Quality Management(QM)に使用する必要があります。Workforce Managementでは、uccxhruserユーザがアクセスできない履歴データテーブルにアクセスする必要があるため、uccxhruserを使用する必要があります。
- **ucxwallboard** – このユーザは、UCCX Engineのメモリから書き込まれたリアルタイム統計のスナップショットを含むリアルタイムデータベーステーブルに対してのみ選択権限を持っています。テーブルRTCSQsSummaryおよびRTICDStatisticsに制限された選択権限は、uccxwallboardユーザを使用して、ウォールボードアプリケーションからソースされる単純で複雑でないクエリを頻繁に使用してUCCXデータベースを照会します。

トラブルシューティング手法

UCCXリリース10.0以降で、**utils uccx database dbperf start <totalHours> <interval>** コマンドを入力し、UCCXデータベースのパフォーマンストレースを開始します。このコマンドのinterval引数はトレースコレクションの間隔を決定し、totalHours引数はトレースが無効になるまでの合計時間を決定します。これらのパラメータはオプションです。コマンドの実行時にこれらを指定しない場合は、デフォルト値の20分と10時間が使用されます。

たとえば、**utils uccx database dbperf start 24 30**コマンドを入力して、データベースのパフォーマンストレースを有効にし、24時間30分ごとにパフォーマンス統計情報のデータを収集します。

CLIコマンドで取得したデータを収集する手順は、コマンド出力に表示されます。

```
admin:utils uccx database dbperf stop
Execution of dbperf has been stopped
Command Successful
admin:utils uccx database dbperf start 24 30
The script runs every 30 minutes over a total duration of 24 hours.
Please collect files after 24 hours

Use "file get activelog uccx/cli/dbperf_171013134928.log" to get the file
Use "file view activelog uccx/cli/dbperf_171013134928.log" to view the file
Command Successful
admin:
```

totalHoursが指定された後、データ・コレクションは自動的に停止します。データ収集を手動で停止するには、utils **uccx database** dbperf stopコマンドを入力して、データ収集を手動で停止します。

```
admin:utils uccx database dbperf stop
Execution of dbperf has been stopped
Command Successful
admin:
```

UCCXバージョンがリリース9.0(2)以前の場合と、 **utils uccxデータベースdbperf** コマンドを使用できません。詳細については、Technical Assistance Center(TAC)にお問い合わせください。

TACは、Cisco Bug ID CSCuc68413に添付されたdbperf.shスクリプトをリモートサポートアカウントに手動で実行します。

スクリプトの実行を手動またはCLIコマンドを使用して開始するタイミングを決定する場合は、間隔と合計時間を考慮して、 uccxoninit プロセスは、根本原因の分析に必要な情報を収集するために、これらの期間に大きく変動するか、または高い状態を維持します。

さらに、定期的にshow process loadコマンドを入力し、dbperfトレーススクリプトで収集されたログを関連付けるために、CPUの変動を確認します。

データ分析

dbperfスクリプトのonstat -g ses 0の実行で収集されたログには、UCCXデータベースに対して発行されたアクティブなクエリが表示されます。uccxoninitプロセスの高いCPU使用率は、通常、実行に時間がかかる複雑なクエリの結果です。目標は、最もリソースを消費するクエリを決定し、それらのクエリのソースクライアントを決定し、クライアントからのクエリを即時解決し、長時間実行されるクエリを最適化して永続的な解決を実現することです。

dbperfスクリプトによって収集されたログでは、CPUの高変動またはuccxoninitプロセスによるCPU高消費の持続を引き起こす可能性が最も高いクエリを探します。

疑わしいクエリ：

- uccxhruserとして接続されたセッションから発行されます。前述のとおり、uccxhruserには、膨大な数の設定テーブルと履歴テーブルから情報を選択する権限があります。その結果、複数のテーブルにまたがる複雑で長いクエリを実行し、UCCXデータベースのパフォーマンスに影響を与える可能性があります。絶対ではありませんが、uccxwallboardとuccxworkforceのユーザはUCCXデータベース内のテーブルへのアクセスを制限します。これらのユーザが発行するパフォーマンスへの影響を引き起こす複雑なクエリはほとんどありません。さらに、UCCXデータベースに対してUCCX Historical Reporting Client(HRC)またはCisco Unified Intelligence Center(CUIC)が発行したuccxhrcareが発行するクエリ。これらのクエリは静的であり、変更することはできません。また、関連するインジケータとともにクエリが書き込まれ、テストされ、パフォーマンスへの影響を最小限に抑えるように調整されます。
- 履歴テーブルに対して集中的なクエリを実行する：UCCXデータベースが複数のテーブル間で結合を実行する必要があるクエリ、大量の情報の選択、インデックスなしのフィールドでの操作は、UCCXデータベースに影響をします。

uccxhruserとして実行されるHRテーブルを含む複雑なクエリの例を次に示します。

```
session #RSAM total used dynamic
id user tty pid hostname threads memory memory explain
435050 uccxhrus WBBOX 836 10.16.5. 1 90112 80712 off

...................

Current SQL statement :
 SELECT x.resourceName, t.eventType, x.datetime, x.extension FROM ( SELECT
 t1.resourceID, t1.resourceName, t1.extension, MAX(t2.eventDateTime) AS
 datetime FROM Resource AS t1, AgentStateDetail AS t2 WHERE t2.agentID
 = t1.resourceID AND t1.assignedTeamID = 21 and t1.active GROUP BY
 t1.resourceID, t1.resourceName, t1.extension ) AS x, AgentStateDetail AS
 t WHERE t.agentID = x.resourceID AND t.eventDateTime = x.datetime
 ORDER BY x.resourceName
```

上記の例は、ホストWBBOXを送信元とするuccxhruserが入力した複雑なクエリを示しています

。このクエリが頻繁に入力された場合、または前のクエリが結果を返す前に定期的にに入力されたされたUCCXデータベースにににパフォーマンスににに影響。

まれに、UCCXデータベースのパフォーマンスが低下する場合があります(および uccxoninit プロセスは、組み込みのパージプロセスの結果として変動するか、高い状態のままになります)。消去プロセスは、データベースのサイズを維持するために、UCCXデータベース内の構成テーブルと履歴テーブルからデータを削除するように設計されています。消去は、データベースのサイズまたはデータベースに含まれる最も古いレコードに基づいてスケジュールできます。

パージプロセスが実行されると、データは1つのクエリで削除されます。削除するレコードの量に基づいて繰り返し実行されることはありません。つまり、削除する必要がある大量のデータがパージによって検出された場合、このデータを削除しようとする際に1つのクエリが発行されます。

UCCX AppAdminページで消去スケジュールまたはパラメータを変更して大量のデータを削除するように消去をスケジュールすると、次にスケジュールされた消去でこの単一のクエリが完了するまでに多大な時間がかかることがあります。したがって、データベースインスタンスのCPU使用率が上がります。

dbperfスクリプトの出力で、パージクエリを確認できます。ユーザーuccxuserが入力した唯一のクエリで、sp_purgeストアドプロシージャを呼び出す必要があります。

```
session #RSAM total used dynamic
id user tty pid hostname threads memory memory explain
5628 uccxuser - -1 CC-EXPR- 1 544768 523408 off



Current SQL statement in procedure db_cra:sp_purge
 proc-counter 0x0x4ccf9260 opcode SQL


delete from contactroutingdetail
where (exists
(select 1
 from contactcalldetail as ccdr
 where (and (and (and (and (and (= contactroutingdetail.sessionid,
ccdr.sessionid), (= contactroutingdetail.nodeid, ccdr.nodeid)),
(= contactroutingdetail.sessionseqnum, ccdr.sessionseqnum)),
(= contactroutingdetail.profileid, ccdr.profileid)), (>= ccdr.enddatetime,
p_purgefrom)), (< ccdr.enddatetime, p_purgeto))));
```

一般的な問題

最近のCisco TACおよびCisco Development Engineeringの経験に基づき、uccxoninitプロセスでCPU使用率が高くなる最も一般的な問題を次に示します。

- エンタープライズ内のクライアントがuccxhruserとして接続し、ウォールボードテーブル（RTICDStatisticsおよびRTCSQsSummary）に結合された複雑なクエリを頻繁に実行して、ウォールボードまたはカスタムレポートソリューションを提供します。ウォールボードを使用する場合は、uccxwallboardユーザーと、リアルタイムテーブルへのクエリを制限してください。ウォールボードまたはウォールボードに類似した周波数の履歴テーブルまたは構成テーブルを照会する機能はサポートされていません。
- クライアントは、セカンダリノードではなく、アクティブなプライマリノードでカスタム履歴レポートを実行しようとします。スタンバイ・ノードで履歴レポートを生成するストアド

・プロシージャ（カスタムまたはデフォルト）のみを実行します。CUICとHRCはデフォルトでスタンバイノードでクエリを実行しますが、カスタム履歴レポートを作成する場合、開発者は、これらのクエリを実行するか、これらのストアドプロシージャを実行するかを選択できます。

- Cisco Workforce Management(WFM)は、startdatetimeフィールドにフィルタをかけるために、ContactRoutingDetailテーブルに複雑なクエリを発行します。このテーブルのこのフィールドにはデフォルトでインデックスが作成されていないため、このクエリのパフォーマンスが低下します。WFMは、UCCXからWFMにデータを同期するために、このクエリを定期的に発行します。この問題は、Cisco Bug ID CSCtz23710に記載されており、WFMリリース9.0(1)SR4で解決されています。この問題が発生した場合は、Cisco Bug ID CSCtz23710の0の0の0に0を0を0の0の0の0を0修正0を0を0として0を0として含0のWFMWFMWFMWFMWFMWFMWFMWFMのバージョンにWFM0をの0をに0を0として0を0に0に0として0として0として0を0として00を0として0を..
- パージしきい値は、次にスケジュールされたパージによって大量のデータが削除されるように変更されます。1回の更新でパージパラメータを大幅に変更するのではなく、パージスケジュールの変更を繰り返し行い、パージ設定の変更の間に数日かかります。これにより、パージプロセスで各パスの小さなデータセットを削除できるため、削除操作のパフォーマンスが向上します。
- DialingListテーブルが非常に大きい。DialingListテーブルには、アウトバウンドキャンペーンにアップロードされたすべての連絡先が保存されます。UCCXリリース8.0および8.5では、数百万のレコードがアウトバウンドキャンペーンにアップロードされた後、パフォーマンスの問題によってテーブルが照会されます（この結果、uccxoninitプロセスでCPU使用率が高くなり、AppAdminページのロードが遅くなります）。 パフォーマンスの問題を軽減するには、DialingListテーブルをクリーンアップするcronジョブスクリプトをインストールするTACケースをオープンします。UCCXリリース9.0では、パフォーマンスを改善するためにAppAdminからより効果的なクエリを実行するために、このテーブルにインデックスが追加されました。この変更により、最も極端なケースを除くすべてのケースでこの問題が解決されました。UCCXリリース10.0では、DialingListが2つのテーブルに分割されています。1つはアクティブな連絡先に対するもので、もう1つは履歴連絡先に対するものです。この問題の包括的な修正が提供されます。