Высокая загрузка ЦП informix

Содержание

[Введение](#)
[Информация о функциональной возможности](#)
[Методология](#)
[Анализ данных](#)
[Типичные неполадки](#)

Введение

Этот документ описывает, как Унифицированный Contact Center Express (UCCX) действия, которые требуют локального доступа к базе данных UCCX, мог бы медленно выполнять. Это заставляет Страницы AppAdmin медленно загружаться, обновления AppAdmin для занимания много времени для взятия влияния, задержки в ответ на запрос wallboard, Менеджер Трудовых ресурсов, чтобы быть неспособным сделать запрос данных UCCX, и другой производительности и проблем со стабильностью.

Загрузка команды show process, введенная в CLI, показывает, что **uccxoninit** использует большое количество ЦП. Процесс **uccxoninit** представляет экземпляр Базы данных Informix UCCX, который работает на сервере UCCX.

Информация о функциональной возможности

Ядром базы данных, которое поддерживает приложение UCCX, является Informix IBM. Конфигурация и исторические сведения, которые добавлены к Странице AppAdmin UCCX и произведены приложением UCCX, сохранены в экземпляре Informix UCCX.

Приложение UCCX предоставляет трех пользователей, которые могут использоваться для доступа к базе данных UCCX непосредственно для извлечения информации в целях wallboard приложений, Управления качеством, менеджмента Трудовых ресурсов и пользовательского исторического создания отчетов.

Сведения о пользователе, разрешения каждого пользователя и намеченная цель каждого пользователя описаны здесь:

- **uccxhruser** - Этот пользователь имеет, выбирают разрешения многим конфигурация и таблицы истории в базе данных UCCX и должен использоваться только для пользовательского исторического создания отчетов и Cisco Унифицированный менеджмент Трудовых ресурсов (WFM). Запросы и сохраненные процедуры, выполняемые этим пользователем, могли бы выполнить сложные, длительные запросы. Из-за профиля типичного исторического создания отчетов или пользователя WFM, эти запросы и сохраненные процедуры не должны часто выполняться, как это произошло бы для wallboard приложения.

Несмотря на то, что много wallboard приложений требуют данных, содержавших в конфигурации и таблицах истории, к которым uccxhruser имеет доступ, это технически не поддерживается для использования этого пользователя для выполнения сложных, частых запросов против UCCXdatabase в целях wallboard приложения.

- **uccxworkforce** - uccxworkforce пользователь имеет доступ к Команде, Ресурсу и таблицам Супервизора и должен использоваться для Cisco Унифицированное Управление качеством (QM). Менеджмент трудовых ресурсов должен использовать uccxhruser, поскольку это требует доступа к таблицам исторических данных, которые не доступны uccxworkforce пользователем.
- **uccxwallboard** - Этот пользователь имеет, выбирают разрешения только на таблицах базы данных в реальном времени, которые содержат снимки статистики в реальном времени, записанной от памяти Механизма UCCX. Избранные разрешения, ограниченные таблицами, RTCSQsSummary и RTICDStatistics имеют в виду uccxwallboard пользователя, должны использоваться для запроса базы данных UCCX часто с простыми, несложными запросами, предназначенными, чтобы быть полученными wallboard приложением.

Методология

В Выпуске 10.0 UCCX и позже, введите **utils uccx, база данных dbperf запускают <totalHours>** команду **<interval>** для начала отслеживания производительности на базе данных UCCX. **Аргумент interval** в этой команде определяет периодичность набора трассировки, и **totalHours** аргумент определяет общее количество времени выполнения отслеживания, прежде чем это будет отключено. Эти параметры являются дополнительными. Если они не заданы, когда команда выполняется, значения по умолчанию 20 минут и 10 часов используются.

Например, введите **utils uccx, база данных dbperf запускаются 24 30** команд, чтобы включить отслеживание производительности на базе данных и собрать данные по статистике производительности каждые 30 минут в течение 24 часов.

Инструкции для сбора данных, полученных командой CLI, распечатаны в выходных данных команды.

```
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:
```

После данного **totalHours** автоматически останавливается сбор данных. Для ручной остановки сбора данных введите **utils uccx, база данных dbperf останавливают** команду.

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

Если версией UCCX является Выпуск 9.0 (2) или ранее и **utils uccx база данных dbperf** команда не доступен, свяжитесь с Центром технической поддержки (TAC) для дальнейшей поддержки.

TAC выполнит dbperf.sh сценарий, подключенный к идентификатору ошибки Cisco CSCuc68413 вручную с Удаленным Доступом к счету Поддержки.

Когда вы определяете, когда запустить выполнение сценария или вручную или через команду CLI, периодичность, и общее время, гарантировать, что ЦП, использованный процессом **uccxoninit**, колеблется значительно или остается высоким в течение тех периодов для сбора необходимой информации для анализа основных причин.

Кроме того, периодически вводите **команду load процесса показа** для определения, когда ЦП колеблется для корреляции журналов, собранных dbperf отслеживание сценария.

Анализ данных

Журналы, собранные выполнением dbperf сценария **onstat-g СЭП 0**, показывают активные запросы, которые выполнены против базы данных UCCX. Высокая загрузка CPU на процессе **uccxoninit**, как правило, является результатом сложных запросов, которые занимают много времени для выполнения. Цель состоит в том, чтобы определить запросы, которые используют большинство ресурсов, определяют исходного клиента для тех запросов, отключают запросы от клиента для немедленного разрешения и оптимизируют длительные запросы для постоянного разрешения.

В журналах, собранных dbperf сценарием, ищите запросы, которые обрабатывает наиболее вероятная причина высокие колебания ЦП или поддержанное потребление высокой загрузки CPU **uccxoninit**.

Подозрительные запросы:

- Выполнены от сеансов, связанных, поскольку **uccxhruser** - Как описано ранее, **uccxhruser** имеет привилегии выбрать информацию из огромного количества конфигурации и таблиц истории. В результате сложные, длительные запросы через множественные таблицы могут быть созданы и могут иметь влияния на производительность на базе данных UCCX. Несмотря на то, что не абсолютный, **у uccxwallboard** и **uccxworkforce** пользователей есть такой ограниченный доступ к таблицам в базе данных UCCX, сложные запросы, которые вызывают влияние на производительность, выполненное этими пользователями, маловероятны. Кроме того, запросы, выполненные **uccxhrcare**, выполненным клиентом исторического отчета (HRC) UCCX или Cisco Unified Intelligence Center (CUIC) против базы данных UCCX. Эти запросы статичны и не могут модифицироваться и запросы, наряду с соответствующим indicies, были записаны, протестированы и настроены для минимального влияния на производительность.

- Выполните интенсивные запросы на таблицах истории - Запросы, которые требуют, чтобы база данных UCCX выполнила множественные соединения через таблицы, выбрала значительные части информации или воздействовала на неиндексируемые поля, мог вызвать влияния на производительность к базе данных UCCX.

Пример со сложным запросом, который включает таблицу HR, выполненную как **uccxhruser**, показывают здесь:

```
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
```

Приведенный выше пример показывает сложный запрос, введенный **uccxhruser**, полученным от хоста **WBBOX**, который мог вызвать влияние на производительность на базе данных UCCX, если бы это часто вводилось или периодически вводилось, прежде чем, предыдущий запрос возвратил результаты.

Несмотря на то, что редкий, производительность базы данных UCCX может также ухудшиться (и загрузка ЦПУ процесса **uccxoninit** колеблется или остается высокой), в результате встроенного процесса чистки. Процесс чистки разработан для удаления данных из конфигурации и таблиц истории в базе данных UCCX для поддержания размера базы данных. Чистка может планироваться на основе размера базы данных или самой старой записи, содержавшей в базе данных.

Когда процесс чистки выполняется, данные удалены с одним запросом. Это не сделано многократно на основе суммы записей для удаления. Это означает, что, если чистка обнаруживает большое количество данных, которые должны быть удалены, это выполняет единый запрос в попытке удалить эти данные.

Модификация списка чистки или параметров от Страницы AppAdmin UCCX для планирования чистки для удаления большого количества данных может заставить этот единый запрос, после следующей запланированной чистки, занимать значительное количество времени для завершения. Поэтому это подвозит загрузку ЦПУ экземпляра базы данных.

В выходных данных dbperf сценария может быть замечен запрос чистки. Это должен быть единственный запрос, введенный пользователем **uccxuser**, который вызывает **sp_purge** сохраненную процедуру.

```
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))));
```

Типичные неполадки

На основе недавнего Центра технической поддержки Cisco и опыта разработчиков Cisco, это обычно замеченные проблемы, которые вызывают высокую загрузку ЦП на процессе **uccxoninit**:

- Клиент на предприятии соединяется как **uccxhruser** и выполнения частые сложные запросы на wallboard таблицах (RTICDStatistics и RTCSQsSummary) присоединенный с таблицами истории для обеспечения wallboard или пользовательского решения для создания отчетов. Для использования wallboard только используйте **uccxwallboard** пользователя и ограничьте запросы оперативными таблицами. Способность сделать запрос исторического или таблиц конфигурации от wallboard или с частотой, подобной wallboard, не поддерживается.
- Клиент пытается выполнить пользовательские отчеты предыстории на активном основном узле вместо вторичного узла. Только выполните сохраненные процедуры, или пользовательские или по умолчанию, которые производят отчеты предыстории на основном узле немеханизма. CUIC и HRC выполняют запросы на неосновном узле по умолчанию, но при разработке пользовательского отчета предыстории, у разработчика есть выбор на который узел выполнить эти запросы или выполнить эти сохраненные процедуры.
- Менеджмент Трудовых ресурсов Cisco (WFM) выполняет сложный запрос на таблице ContactRoutingDetail, чтобы попытаться фильтровать на startdatetime поле. Никакой индекс не создан на этом поле в этой таблице по умолчанию, таким образом, производительность этого запроса плоха. WFM периодически выполняет этот запрос в попытке синхронизировать данные от UCCX до WFM. Этот вопрос перехвачен в идентификаторе ошибки Cisco CSCtz23710 и решен в Выпуске 9.0 (1) SR4 WFM. Клиенты, которые испытывают эту проблему, должны обновить к версии WFM, который содержит исправление для идентификатора ошибки Cisco CSCtz23710.
- Пороги чистки модифицируются таким образом, что следующая запланированная чистка пытается удалить большое количество данных. Вместо того, чтобы значительно модифицировать параметры чистки в одиночном обновлении, модификации списка чистки сделаны многократно с несколькими днями между изменениями конфигурации чистки. Это позволяет процессу чистки удалять меньшие наборы данных в каждом проходе, который улучшает производительность удалить операции.
- Таблица DialingList является чрезвычайно большой. Таблица DialingList хранит все контакты, загруженные к Исходящим Кампаниям. В Версиях 8.0 и 8.5 UCCX, после того, как миллионы записей загружены к Исходящим Кампаниям, результат проблем производительности тогда таблица делают запрос (который вызывает высокую загрузку

CPU на процессе uccxoninit и медленной загрузке Страницы AppAdmin). Для смягчения проблем производительности откройте кэйс TAC (Центра технической поддержки) для установки сценария работы крона, который очищает таблицу DialingList. В Выпуске 9.0 UCCX индекс был добавлен к этой таблице для более эффективных запросов от AppAdmin в попытке улучшить производительность. Это изменение решило вопрос во всех кроме наиболее наихудших ситуаций. В Выпуске 10.0 UCCX DialingList был разделен на две таблицы, один для активных контактов и другого для исторических контактов, который предоставляет всестороннее исправление для этой проблемы.