

NSOが高いCPUを消費する場合のデータ収集

内容

[概要](#)

[前提条件](#)

[要件](#)

[使用するコンポーネント](#)

[背景説明](#)

[収集するデータ](#)

[追加情報](#)

[関連情報](#)

概要

このドキュメントでは、CPU使用率が100 ~ 150 %に増加した場合に必要なNetwork Services Orchestrator(NSO)データ収集について説明します。

前提条件

要件

このドキュメントに関する固有の要件はありません。

使用するコンポーネント

このドキュメントの内容は、特定のソフトウェアやハードウェアのバージョンに限定されるものではありません。

このドキュメントの情報は、特定のラボ環境にあるデバイスに基づいて作成されました。このドキュメントで使用するすべてのデバイスは、クリアな(デフォルト)設定で作業を開始しています。本稼働中のネットワークでは、各コマンドによって起こる可能性がある影響を十分確認してください。

背景説明


NBから複数のトランザクションが処理されると、NSOのCPU消費量は通常消費量の約100 ~ 150 %に増加します。この場合、CPUパフォーマンスを低下させる原因を見つける必要があります。同時に、NSOはRESTCONF(使用されている場合)クエリに正しく応答しません。

この記事では、問題を適切にトラブルシューティングし、いくつかの解決策を提案できるように、問題中に収集する必要があるすべての重要なデータを上げます。

収集するデータ

Linuxの観点から：

- LSCPU
- top
- フリー - h
- vmstat
- cat /proc/meminfo
- pstree -c
- PSオクソ |並べ替え

 注：これらの詳細（「lscpu」を除く）を定期的にキャプチャして、要求がNBから到達した場合のシステムの動作を理解できます。

NSOの観点：

- ncs : ステータス | grep ロック
- 進行状況トレースを有効にします。

```
admin@ncs(config)# commit dry-run
```

```
cli {
```

```
  ローカルノード{
```

```
    データの進行状況{
```

```
      + trace all {
```

```
        +宛先{
```

```
          + file progress-all.txt;
```

```
          +ログのフォーマット ;
```

```
        +}
```

```
      +}
```

```
    }
```

```
  }
```

```
}
```

```
admin@ncs(config)# commit
```

- 次の情報を「n」秒ごとにキャプチャします（スクリプトとして実行可能）。

```
シーケンス=0
ncs —status >& /dev/null;
ncs —debug-dump ncs.dd.$((seq++));
ncs —status > ncs.stat.$((seq++));
スリープ30、#Configured according ユーザへ
done
```

次に、この問題を軽減するために実行できる改善策をいくつか示します。

1. 次のようにセッション数を制限します (現在このセットはありません)。

<セッション制限>

<セッション制限>

<context>rest</context>

<max-sessions>100</max-sessions>

</session-limit>

</session-limits>

- b. NSOプロセスが何らかの原因で終了したかどうかを確認し、終了した場合はaudit.logに記録するように監査ルールを有効にします。

```
sudo auditctl -a exit,always -F arch=b64 -S kill -k audit_kill
```

トラブルシューティングと分析を行うには、audit.log、devel.log (level=traceに設定することが望ましい)、ncs-java-vm.log、およびNBのログに加えて、前述の詳細情報が必要です。

追加情報

- Q. NSOはNBアプリケーションからのRESTCONF要求を実際にどのように処理しますか。

A. ノースバウンドアプリケーションがRESTCONF要求を送信すると、NSOに基づく一意のトランザクションとして扱われます。つまり、NSOはCDB全体をロックでき、現在のトランザクションが完了するまで他のトランザクションを許可しません。これを行うと、NSOのトランザクションの性質が維持され、問題が発生した場合にロールバックを実行できます。

NSOコミットキューは、後続のトランザクション要求が完了するたびに処理できます。また、開始または完了するたびにdevel.log内のトランザクションロックを追跡できます。大量のクエリが実行される場合、NSOに大量のオーバーヘッドが発生し、トランザクションが予想よりも長い時間コミットキューに入ります。RESTCONF要求がグループ化された場合、トランザクションオーバーヘッドが減少するため、スループットが増加します。また、NSOは、1つのトランザクション内で同時に可能な限り多くのことを実行できます。たとえば、トランザクションに2つのデバイス設定変更が含まれている場合、NSOはCDBをロックし、両方のデバイスに同時にアクセスして編集し、トランザクションを完了できます。これは、それぞれ1つのデバイスを含み、両方が変更される2つのトランザクションとは異なります。NSOは最初のトランザクションのCDBをロック

し、最初のデバイスを編集し、トランザクションを完了してから、2番目のデバイスに同同の手順を実行できます。

関連情報

- [シスコテクニカルサポートおよびダウンロード](#)

翻訳について

シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。