Hewlett Packard Enterprise

HP-UX 11i カーネル・チューニング技法

HP-UX 11i のカーネル・パラメータには、個々のシステムの用途に合わせてチューニング可能なものが数多くあります。これらの値を調整することで、パフォーマンスを改善したり、リソースをより効果的に割り当てたりできます。例えば「システムあたりのプロセス数」や「ユーザあたりのプロセス数」を適切に設定すれば、リソースを各ユーザに効率的に配分し、システム全体を最適な状態に保てます。

《連載期間：2005 年 2 月～2005 年 3 月》

―目次―

第 1 回

プロセスとスレッドの管理

2005 年 2 月

HP-UX 11i のカーネル・パラメータには、個々のシステムの用途に合わせてチューニング可能なものが数多くあります。これらの値を調整することで、パフォーマンスを改善したり、リソースをより効果的に割り当てたりできます。例えば「システムあたりのプロセス数」や「ユーザあたりのプロセス数」を適切に設定すれば、リソースを各ユーザに効率的に配分し、システム全体を最適な状態に保てます。連載第 1 回では、HP-UX におけるプロセスとスレッドの管理にかかわるチューニング・ポイントを説明します。

カーネル・パラメータの役割

HP-UX 11i のカーネル・パラメータには、個々のシステムの用途に合わせてチューニング可能なものが数多くあります。これらの値を調整することで、パフォーマンスを改善したり、リソースをより効果的に割り当てたりできます。各パラメータに設定すべき値は、ハードウェア構成やアプリケーションの組み合わせなどに依存するため、システムによって千差万別です。それぞれには適切なデフォルト値があらかじめ設定されていますが、実際に使用するユーザのニーズに合わせて変更することで、HP-UX の能力をフルに引き出すことが可能です。
カーネル・パラメータの多くは、個々のサブシステムの挙動に対応しており、おもに以下の種類に分けることができます。

- プロセス管理
- メモリ・ページング
- ファイル・システムと LVM
- FibreChannel
- 非同期 I/O
- キャラクタ・モード I/O
- System V プロセス間通信
- アカウンティング・ログ
- カーネル・ダンプ
- その他

本連載では、これらのうち主要なカーネル・パラメータについて、その使い方と機能を紹介します。

なお、HP-UX のカーネル・パラメータを変更する手段としては、コマンド kctune および GUI ツール kcweb が利用可能です。

チューニングを始める前の注意点

カーネル・パラメータには、それぞれが独立して作用するものや、相互に影響し合うものがあります。不適切な値を設定したり、複数のパラメータの値を互いに不適切な組み合わせで設定したりすると、リソースの不足や競合などにより、動作上の問題が発生する可能性があります。こうした問題は、後から原因を究明するのが困難になりがちです。よってカーネル・パラメータを変更するときには、最初にその変更がシステム全体にどのような影響を与えるかを十分に検討してください。
また各パラメータは、必ず許容範囲内の値に設定します。管理ツール SAM を使ってパラメータを調整する方法であれば、許容範囲外の値は拒否されます。さらに、複数のパラメータが互いに影響し合うケースも少なくないので、これらの値は相互のバランスを保ちながら設定していく必要があります。

本稿を含め、カーネル・パラメータの設定方法について記したドキュメントには、その時点で正確と考えられる情報が記載されています。しかしシステム構成の変更などにより、実際のシステムにおけるデフォルト値や制限値との間に食い違いが生じる可能性もあります。そうした場合は、HP-UX の/etc/conf/master.d ディレクトリに格納されているファイルを参照し、実際のシステムにおける値を確認してください。(HP-UX11i v1 の場合)

プロセス管理のチューニング

今回は、HP-UX におけるプロセスとスレッドの管理にかかわるチューニング・ポイントを説明します。例えば「システムあたりのプロセス数」や「ユーザあたりのプロセス数」を適切に設定すれば、リソースを各ユーザに効率的に配分し、システム全体を最適な状態に保つことができます。

プロセスの最大許容数

プロセスの最大許容数として適正な値は、個々のシステムによって大きく異なります。例えば、CDE 環境や Motif 環境が稼動している GUI ベースのワークステーションでは、1 人のユーザが 100 以上のプロセスを同時に実行することもあります。一方、ダム端末が接続された大規模なマルチユーザ・システムの場合、各ユーザが実行するプロセスの数は 2〜3 個であっても、端末数が数 100 台に及ぶかもしれません。この 2 つの対照的な例を見ても分かるように、プロセス数はシステムの構成によって大きく変化します。

プロセスの最大許容数を管理するカーネル・パラメータとしては、以下の 2 つがあります。

セキュリティの確保

- **nproc**
  システム全体で同時に実行可能な最大プロセス数を定義します。この数には、remsh などのネットワーク・コマンドを通じて他のシステムから起動されたリモート実行プロセスも含まれます。nproc の値は、システムのピーク時に合わせて、同時にログインするすべてのユーザに対して十分な数のプロセスを提供できるように設定する必要があります。

- **maxuprc**
  個々のユーザが同時に実行可能な最大プロセス数を定義します。この数には、ログイン・シェル、ユーザ・インターフェイス・プロセス、実行中のプログラムと子プロセス、I/O プロセスなどが含まれます。
  X-window、CDE、および Motif では、1 人のユーザが同じログイン名（ユーザ ID）を使って同時に複数のログインするケースがあります。このような場合は、プロセスが所属するプロセス・グループの違いにかかわらず、すべてのプロセスの個数が合算されます。プロセスが親プロセス・グループから切り離された場合、そのプロセスはカウントされなくなります（印刷ジョブや特定の専用アプリケーションなど）。

  maxuprc の値は、すべてのユーザについて、通常時のニーズを満たせるように設定します。あまり大きな値を設定すると、プログラムの暴走などによって過度に多数のプロセスが生成されたとき、他のユーザが新しいプロセスを生成できなくなる可能性が生じます。またシステムを悪用するユーザがいた場合、他のユーザを保護できなくなります。

カーネル・スレッドの最大許容数

複数のプロセッサーを備えたシステムでは、プロセス内部の複数のスレッドが、それぞれ別々のプロセッサー上で同時に実行されます。こうしたスレッドによるシステム・リソースの消費は、以下の 2 つのカーネル・パラメータを使って管理できます。

- **max_thread_proc**
  1 つのプロセスが作成して同時に実行するスレッドの最大許容数を定義します。

  プロセスが複数のスレッドを生成すると、プロセス空間の一部がスレッドごとに複製されるため、一定のシステム・リソースとメモリを消費します。もし暴走したプロセスや悪意のあるユーザが大量のスレッドを生成すると、システム性能が著しく低下したり、他の誤動作が誘発されたりするおそれがあります。max_thread_proc を適切に設定することで、プロセスが異常な数のスレッドを生成した場合でも、システム・リソースが過剰に消費されるのを防止できます。

  max_thread_proc に値を指定する場合は、システム上で運用するマルチスレッド・アプリケーションについて、もっとも多くのスレッドが生成される条件でのスレッド数を調査します。max_thread_proc には、少なくともこの数を上回る値を設定します。ただし、上述したようなトラブル時でも他のプロセスを保護できるように、あまり大きな値に設定してはいけません。

- **nkthread**
  システム上に同時に存在するカーネル・スレッドの最大許容数を定義します。
  大規模なマルチスレッド・アプリケーションでは、大量のスレッドが生成されることがあります。1 つのプロセスが作成できるスレッド数は上述のパラメータ max_thread_proc で制限可能ですが、そうしたプロセスがシステム上にいくつも起動するケースも考えられます。

  nkthread は、システム上のスレッド総数を制限します。このパラメータ値により、異常な数のスレッドによってシステムが過負荷状態となることを防げます。例えば、悪意のあるユーザが故意に大量のマルチスレッド・プログラムを起動し、リソースを枯渇させるような行為に対してシステムを保護します。

  nkthread のデフォルトでは、1 プロセスにつき平均 2 個のスレッドと、システム自体で数個のスレッドを使用可能な値が設定されています。より多くのスレッドが必要な場合は、以下の手順を実施します。

1. システム上の大規模なマルチスレッド・アプリケーションが生成するスレッド数を調べます
2. それらのアプリケーションが同時に実行される場合、それぞれのスレッド数を合計します
3. スレッド数合計に、他のユーザやプロセス用に確保しておく適切なスレッド数を追加します（nproc*2 の値を使用すると便利です）。
4. 上記で算出した総スレッド数よりも十分に大きい値で、かつシステムの完全性に影響を与えない範囲の値を nkthread に設定します。

同時にいくつものマルチスレッド・アプリケーションを運用する大規模システムでは、HP-UX のプロセス管理ツールなどを活用し、システムを試験的に運用させながらカーネル・チューニングを施す必要があるでしょう。

次回は、CPU 時間やメモリの割り当てのチューニング方法について紹介する予定です。

最終回

メモリ・ページングのチューニング

2005 年 3 月

前回は、HP-UX におけるプロセスとスレッドの管理にかかわるチューニング・ポイントを説明しました。今回は、メモリ・ページングに関するカーネル・パラメータを取り上げ、仮想メモリの振る舞いや割り当てを設定する方法について紹介します。

メモリ・ページングのパラメータ

メモリ・ページングに関するカーネル・パラメータを調整することで、仮想メモリ(スワップ・スペース)の振る舞いや制限を設定できます。これらのパラメータは、以下のカテゴリに分類できます。

- システム・スワップの総量
- デバイス・スワップ
- ファイル・システム・スワップ
- 擬似スワップ
- 可変ページ・サイズ

今回は、これらのカテゴリの各パラメータについて、その使い方を説明します。

システム・スワップの総量

システム全体に存在するスワップ・スペースの総量は、デバイス・スワップ、ファイル・システム・スワップ、およびリモート NFS スワップのそれぞれの合計値に相当します。その最大許容値は、maxswapchunks と swchunk という 2 つのカーネル・パラメータによって設定できます。これらのうち、maxswapchunks は、システム全体で作成もしくは割り当て可能なスワッ

プ・チャンクの数を指定します。一方 swchunk は、複数のデバイスまたは複数のファイル・システムにまたがって作成される各チャンクのサイズを決定します。

ただし、これらのパラメータを適切に設定するには、カーネルの動作とシステム内部に関する高度な知識が必要です。詳しい知識がない場合は、swchunk をデフォルト以外の値に変更しないでください。

デバイス・スワップ

ディスク上の一部に設けられたスワップ領域を、デバイス・スワップと呼びます。デバイス・スワップに関連するカーネル・パラメータは、nswapdev の 1 つだけです。このパラメータには、デバイス・スワップに用いるデバイスの数を指定します。これにより、カーネル内部では、指定された数のデバイスを管理するデータ構造(1 つあたり 50 バイト未満)が確保されます。この数を超えるデバイスにはアクセスできません。

ファイル・システム・スワップ

ファイル・システム上に作成されたスワップ領域は、ファイル・システム・スワップと呼びます。この方式では、ページング対象のデータをディスクに直接転送する代わりに、ディスク上のファイルに読み書きします。このため、ファイル・システム・スワップの動作は、デバイス・スワップよりも遅くなります。

ファイル・システム・スワップをサポート可能なファイル・システムの数は、nswapfs で設定できます(このパラメータは、デバイス・スワップにおける nswapdev パラメータに相当します)。nswapfs は、「常時マウントされ、ファイル・システム・スワップに使用するファイル・システム数」に一致するように設定します。なおカーネル内部では、指定された値のファイル・システムを管理するデータ構造(1 つあたり 300 バイト)が確保されます。

ファイル・システム・スワップのパラメータとしては、さらに allocate_fs_swapmap パラメータが用意されています。ファイル・システム・スワップでは、通常、malloc()が呼び出された時点で初めてスワップ・スペースが割り当てられます。しかし、ファイル・システムが満杯状態のときは、エラーが発生することがあります。そこで allocate_fs_swapmap パラメータを使うことで、事前に swapon()が呼び出された時点でスワップ・スペースを割り当てできます。これにより、malloc()実行時のファイル・システム・スワップのエラー発生を防止でき、可用性が向上します。

擬似スワップ

割り当て可能なメモリの大きさは、ディスクおよびファイル・システム上で利用可能なスワップ・スペースの大きさで決まります。しかし、大量の RAM をインストールした大規模なシステムの場合は、これが効率性の低下を招くことがあります。

例えば、200 MB の RAM が搭載されているシステムのルート・ディスクに 1 GB のスワップ・スペースがある場合を考えてみましょう。このようなシステムがシングル・ユーザー・モードで動作しており、なおかつ RAM の 10%程度しか使用しない場合は、ルート・ディスクの 1 GB 分をスワップとして使用するのは非効率です。

こうしたとき、擬似スワップを使用します。RAM のうち使用されていない領域をスワップ領域として割り当てることが可能になり、より大規模で高負荷なプロセスを実行できるようになります。ただし、搭載メモリ量やスワップ・スペースのサイズが小さなワークステーションなどでは、擬似スワップによる効果はほとんど得られません。

擬似スワップの割り当てを使用可能もしくは使用不可能に設定するには、swapmem_on を使います。

パラメータの一覧

allocate_fs_swapmap	ファイル・システム・スワップ・スペースの割り当てを、malloc()呼び出し時ではなく、swapon()呼び出し時に実施するかどうかを指定します。この設定により、スワップ・スペースが不足する可能性が少なくなります。この割り当てを可能に設定するのは、おもに高可用性が重視される場合です。
maxswapchunks	デバイス・スワップとファイル・システム・スワップのチャンク数の最大値を決定します。このパラメータの値、swchunk、そしてシステム・ブロック・サイズ(DEV_BSIZE)を掛けた値がスワップ・スペースの最大サイズを表します。 【注】このカーネルパラメータは HP-UX11i v2 より使われなくなりました。
nswapdev	デバイス・スワップに使用するデバイスの最大許容数。実際のシステム構成に一致する値を設定してください。
nswapfs	ファイル・システム・スワップに使用可能なファイル・システムの最大数。実際のシステム構成に一致する値を設定してください。
page_text_to_local	クラスタ・クライアント上で、プログラムのテキスト・セグメントをローカル・スワップ・デバイスにスワップするかどうかを設定します。有効にした場合は、メモリに新しい内容をロードするためのロード時間は増加します(テキストを破棄してサーバーから再ロードするのではなく、ローカル・ディスクに書き込みます)。一方で、サーバーからクライアントへのネットワーク・トラフィックは減少します。 【注】このカーネルパラメータは HP-UX11i v2 より使われなくなりました。
remote_nfs_swap	リモート NFS ファイル・システムをスワップとして使用するか否かを指定します。クラスタ・クライアントにおいて、NFS マウントされたファイル・システムへのスワップを行うときに使用します。
swapmem_on	物理メモリ上にスワップ領域を確保する擬似スワップ機能を使用するか否かを指定します。
swchunk	スワップのチャンク・サイズを指定します。2 の累乗となる整数値を指定します。

可変ページ・サイズに関するパラメータ

vps_ceiling	ユーザーがページ・サイズを指定しなかった場合にシステムが自動的に選択するページ・サイズの最大値を定義します。
vps_chatr_ceiling	ユーザーがプログラム内で chatr コマンドを使って指定できるページ・サイズの最大値を定義します。
vps_pagesize	ページ・サイズが chatr を使って指定されていない場合に適用するデフォルトのユーザー・ページ・サイズの最小値を定義します。

HP-UX

www.hpe.com/jp/hpux

Hewlett Packard Enterprise