完全ガイド

エンドユーザーコンピューティングと仮想デスクトップインフラストラクチャーソリューション

NUTANIX™

著者について

本書の著者であるBrian Suhrは、エンタープライズ向けインフラストラクチャーの設計や構築、そして運用管理など、IT分野において20年以上の経験を有しています。仮想化やデータセンター、クラウドベースのシステム導入など、様々な分野におけるアーキテクチャーやエンジニアリングに関する専門性を発揮しながら、ハイパフォーマンスな技術チームと共にグローバル規模の業務案件に取り組んでいます。また、DataCenterZombieやVirtualizeTipsなどの著者として、仮想化や自動化、さらにインフラストラクチャーに関する記事コンテンツの執筆に注力し、テクノロジーコミュニティで有益となる製品やサービスに対するエバンジェリストといった存在になっています。Twitter：@bsuhrでBrian Suhrをフォローいただけます。

公式リリース"1.0"

目 次

本eBOOKについて

本書では、VDIおよびエンドユーザーコンピューティング（EUC）のための
インフラストラクチャーデザインに焦点を絞って解説を行います。また、
掲載コンテンツについては、近々出版予定の「Architecting and Designing
End-User Computing Solutions」でインフラストラクチャーについて
解説している章の内容を活用しています。

はじめに

EUCサービスやアプリケーションを提供するために、適切な戦略とソフトウェアベンダーを選定した後、次の重要な意思決定ポイントとなるのが、アプリケーションやデスクトップの仮想化プロジェクトに不可欠なインフラストラクチャーの選択です。

システムデザインの前提に、安定的で高可用性を持ち、高いパフォーマンスを発揮するインフラストラクチャーが据えられていない場合、IT部門は、EUCプロジェクトの導入や運用フェーズに入った段階で、多くの問題に直面することになります。問題に直面してからインフラストラクチャーの重要性に気付くようでは既に手遅れで、膨大なIT部門の作業時間がインフラストラクチャーのために浪費されることになります。アーキテクトやエンジニアは、インフラ対応にその時間を割くことなく、EUCサービスやアプリケーションの提供に集中できなければなりません。

EUCのためのインフラストラクチャーをデザインする過程で考慮すべき、幾つかの重要な要素があります。組織が持つ要件に沿った形でこれらの要素を使用することで、どんなアーキテクチャーを使って構成するかという点について、より優れた判断を下すことができます。EUCプロジェクトにおけるアーキテクチャーやベンダーを評価する際、考慮すべき対象としては、以下のような要素があります：

- エントリーポイント
- 拡張性
- パフォーマンス
- モニタリング
- キャパシティ

エントリーポイント

多くの場合、インフラストラクチャーの導入開始時期や検討開始時期は、プロジェクトを実施するか否かの分岐点になります。これは、導入開始時点での規模の違いによって、アプリケーションやデスクトップの仮想化と導入を開始するために必要となるインフラストラクチャーおよびそのコストが決まることを意味します。

当初5,000ユーザーでスタートし、最終的に10,000ユーザーまで拡張しようというプロジェクトを計画した場合、初期コストに驚くことはあまり無いでしょう。選定するインフラストラクチャーのタイプにも依存しますが、数千ユーザーの導入になるまで、1ユーザーあたりのコストはあまり意味を成さないからです。

逆に、最終的には10,000ユーザーの導入を目指すものの、当初は500ユーザーでスタートし、プロジェクト実施計画に従い、徐々に拡張していく場合もあります。当初から大規模に導入行なう場合と、500ユーザー規模でスタートする場合に必要な初期のインフラストラクチャーのコストは大差が無いように思われます。当初から大規模な導入を行った場合には、システム環境を拡張しても1ユーザーあたりのコストは安定的で、規模の効果もあって、そのコストは非常に低く見えます。

インフラストラクチャーのコストを決定する上で、1ユーザーあたりのコストは曖昧かつ不透明なものであるにも関わらず、経営層にプロジェクトの判断を仰ぐ場合や、インフラストラクチャーの選定を責任者に委ねる場合には、該当コストの提示が求められます。初期ユーザーコストが高額なインフラストラクチャーを選定する場合、その詳細な理由を説明できなければなりません。ソリューションの評価にあたっては、より自社の環境に適したものが対象となるべきです。そうでない場合には、どれだけのコストを投入するのかという意思決定を下すための準備が必要です。この2つのシナリオを図1に例示しました。

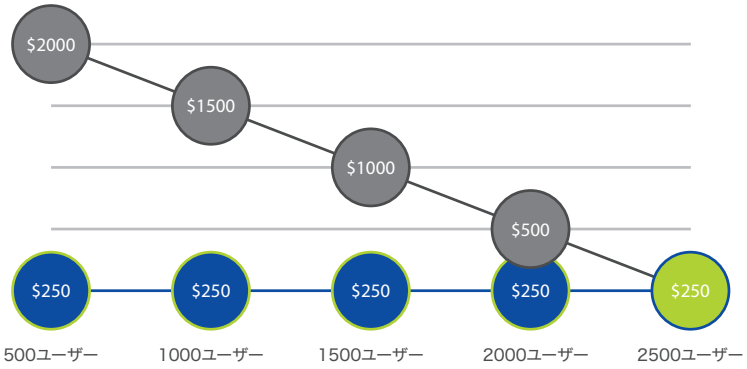

図 1:
デスクトップあたりのエントリーポイント

拡張性

プロジェクトの実行可能性を評価する場合には、アーキテクチャーの拡張性が重要な要素となります。アーキテクトは、複数の異なる案におけるスタート規模を理解しておく必要があります。これは前述のエントリーポイントに遡る話です。つまり、そのアーキテクチャーは、要件に応じた小規模なスタートが可能か？逆に、想定に反し、導入開始時点の規模では足りなくなり、インフラストラクチャーを増設する必要はないか？そして、一定規模まで導入が進まなければ、すべてのリソースを有効活用できないのか？といった内容になります。

どの程度の規模でスモールスタートが可能かという点とは別に、その拡張性もまた非常に重要な考慮点となります。500ユーザーでスタートして10,000ユーザーまで拡張できることを考えるとしたら、該当アーキテクチャーはこの両極に耐えられるものでしょうか？組織は、低い方、高い方いずれならば満足できるでしょう？あるいはその両方でしょうか？

拡張性は、ストレージだけに関わる問題ではありません。サーバーやネットワーク、さらにシステムデザイン上の他のレイヤについても同様となります。1ホストサーバーあたりのVM集約率を抑えるために、サーバーレイヤの構成を調整した場合、拡張に際して、他のデザイン部分にどんな影響が及ぶでしょうか？例えば、当初ホスト1台あたりのメモリを128GBで設計し、最終的に256GB、あるいはそれ以上に拡張する場合、このような将来的な拡張にも対応できるよう、適切なサイズのDIMM (Dual Inline Memory Module)が使用されていることを確認する必要があります。コスト削減を考え、誤った選択をすると、DIMMの制約から、稼動可能なVM数に制限が生じたり、再利用できないDIMMが発生することで、長期的にコストが増加するような事態となります。

アーキテクトは、どの程度のスモールスタートが可能かという点と共に、最大拡張可能規模についても注意を払う必要があります。導入における拡張規模に応じて、開始時点から最終時点までの間には多くの「拡張ポイント」が存在するため、これら全ての中間点も無視することはできません。プロジェクトで定めたユーザー数単位で、導入スケジュールや許容値を上回ることなく、容易に拡張できる方法を見つけることが理想的です。拡張ユーザーの単位は、100～200人程度が適切でしょう。これを超える規模の構成で導入を行う場合には、コストや導入に与える影響を十分に理解しておく必要があります。

パフォーマンス

エンドユーザーエクスペリエンスの良し悪しで評価されるEUCのパフォーマンスには、常に注意を払う必要があります。プロジェクトのあらゆるフェーズで、求められる要件を満足できるようなアーキテクチャーを選択する必要があります。しかし、これはアーキテクチャーによっては厄介な問題となります。スタート段階でのユーザー要件を満足するようにソリューションをサイジングした場合、リニアな拡張性能が得られなければ、将来的にはパフォーマンス上の問題が発生します。アーキテクトは、スモールスタートを選択することで、ソリューションが発揮し得る最大パフォーマンスを犠牲にするような、アーキテクチャー上の妥協はしたくないものです。事前に十分な時間を割いて正しい決定を下すことで、このような問題を避けることができます。

通常、EUCソリューションの設計にあたっては、様々なパフォーマンス要件が存在します。このため、単独でこれら全てのパフォーマンス要件を満足できるフレキシブルなアーキテクチャーを選択する必要があります。複数のタイプのEUCサービスを提供する場合でも、また、アプリケーションやデスクトップの仮想化に限定される場合であっても、複数の異なるパフォーマンス要件に対応できるデザインとなっている必要があります。該当アーキテクチャーが、個別のパフォーマンス要件にどれだけ対応できるか、あるいはできないかを理解していることが、アーキテクチャーの評価や設計プロセスに大きな影響を及ぼします。

キャパシティ

キャパシティの検討も、パフォーマンスの検討と同様になります。EUCの設計にあたっては、対応すべき様々なキャパシティ要件が存在します。該当ソリューションは、サーバーVMやデスクトップVM、そしてアプリケーションを稼動させ、アーキテクチャーのタイプに応じたユーザープロファイルやユーザーデータを維持しなければなりません。デザインの各レイヤには、それぞれ異なるキャパシティ要件が存在します。重複排除が非常に効果的となる大規模データを使用するレイヤが存在する一方で、ユーザー毎に、圧縮可能で非常に小さなサイズでありながら、数千ユーザー分となることで結果的に大きな割合を占めるユーザープロファイルやデータも存在します。

過去における大きな過ちは、過大な、あるいは過少なキャパシティを導入して、求められるパフォーマンスレベルを達成しようしてきたことです。対象となるアーキテクチャーが、必要なキャパシティをどのようにして確保できるか、また、最低限のパフォーマンス要件を確実に満足できるかという点を、設計フェーズで十分に検討する必要があります。該当アーキテクチャーは、ストレージパフォーマンス要件の2〜3倍ものキャパシティを提供すべきではありませし、また、キャパシティ要件を満足するために過剰なパフォーマンスを追加すべきでもありません。理想的なソリューションとは、パフォーマンスやキャパシティを、いずれもかけ離れたものとして扱うのではなく、同じ割合で拡張していけるような柔軟性を持っているものです。

過去、このトピックについて様々な議論や論争が展開されました。多くのユーザーが、パフォーマンスよりキャパシティを優先的に拡張し、パフォーマンスとキャパシティプラニングの問題に陥っています。該当ソリューションに5テラバイトの空き領域があるからといって、500ユーザーを追加できるというものではありません。このシナリオでは、パフォーマンスに大きな問題が発生します。しかし、システム管理者やIT部門長は、ソリューションを拡張すると、なぜこの罠に陥ってしまうのか、未だ正確に理解できていないのです。

モニタリング

モニタリングは、非常に重要でありながら、見落とされるケースが多い要素です。EUC環境でインフラストラクチャーをモニタリングする場合、通常、システム管理者は、パフォーマンスに注目します。管理者には、正常状態を見極め、問題の発生を判断できる能力が要求されます。

モニタリング機能は簡単に利用でき、有益で非常に詳細な情報を提供してくれます。これは、多くのベンダーにあてはまるわけではないので、（ユーザー）エクスペリエンスのモニタリングについては、十分に比較検討する必要があります。

もう1つの要件は、仮想マシンレベルでパフォーマンスをモニタリングができることです。残念ながら、ほとんどのインフラストラクチャーベンダーが、仮想化環境のパフォーマンスを、VMレベルで可視化する機能を提供できていません。ストレージレイヤを迅速に調査し、パフォーマンスの問題が、全体的なレベルなのか、ホストやVMグループなのか、あるいは特定のVMだけで発生している問題なのかなどを特定できる機能は、もはや必須です。

VMレベルでのストレージパフォーマンス管理機能により、ホストレベルのVMにおけるCPUやメモリのパフォーマンス管理でも、同様なアプローチを取ることができます。システム管理者は、VMが一時的に追加パフォーマンスを消費しているのか、あるいは、平均的なユーザーに比べて、定常的にストレージパフォーマンスを消費しているのかを見極めておく必要があります。これにより、パフォーマンスが急激に悪化した場合、その問題の特定に向け、いつ何を詳細に調査すればよいかという判断ができます。

ビルディングブロック

ビルディングブロックとは、事前に構成されたインフラストラクチャーの集合であり、特定の量のリソースやユーザー数に対応できるようになっています。これは、エンドユーザーコンピューティングのためのインフラストラクチャー設計に向けた最も優れたアプローチの1つです。

このアプローチを採用することで、事前に想定が可能なコストやパフォーマンス、キャパシティの拡張モデルを備えたアーキテクチャーを構築することが可能となります。ビルディングブロックのサイズを決定する際、ユーザー数の拡大に向け何を拡張しなければならないのか、また、それをインフラストラクチャーがどのように実装できるかといった点について理解する必要があります。例えば、50ユーザーを100ユーザーに拡張しようとしても、インフラストラクチャーがこのような小規模な拡張に上手く対応できないことがあります。この場合、500や1,000ユーザー規模にまで拡張せざるを得ないかもしれません。インフラストラクチャーを大きなブロック単位でしか拡張できない場合には、その単位に従い、ユーザー数を増やすか、または、インフラストラクチャー拡張のコストが、ユーザー導入単位とは一致しないという事実を受け入れるしかありません。つまり、わずか50や100程度のユーザー数の増加に対応するために、1,000ユーザーに向けたブロック単位で、インフラストラクチャーを購入することになってしまうのです。

これは正に、小規模なユーザー数をサポートするために、大規模なブロックを導入することに他ならず、仮想デスクトップやユーザーセッションのコストを割高に見せます。そして、このようなコストが均一化され、妥当なものとなるのは、計画した全ユーザー数に達してからです。

ビルディングブロック方式のアーキテクチャーは、あらゆるデザインのプロジェクトに対して有効ですが、EUCの導入に際しては、一般に同類のユーザーや同じ特性を持ったユースケースが存在し、グループ単位で導入されます。100ユーザー対応のブロックサイズで進める場合、100ユーザーで必要となるリソースが理解できれば、インフラストラクチャーのブロックによって、確実にこのようなユーザー要件に対応することができます。

仮に、ユーザーが定常運用の状態で15 IOPSと30GBのストレージ容量、2GBメモリ、200MHzのCPU能力を必要とする場合、アーキテクトは、該当ビルディングブロックについて、1,500 IOPS、3TBのディスク容量、200GBのメモリ、20GGHzのCPUが必要であると判断できます。アーキテクトは、リソースに余裕をもってビルディングブロックをデザインすることもできますが、いずれの数値も要件を下回ってはなりません。また逆に、実際に使用されることがない無駄なリソースを各ブロックに追加するようなことは避けるべきです。

このようなデザインにおけるアプローチや粒度を維持することで、50や100ユーザーといった小規模なグループにも対応した拡張が可能となります。つまり、導入やパフォーマンス、キャパシティ、コストに対する計画を、事前に想定した通り段階的かつ着実に遂行することができます。もし、短期間に大規模な導入を望む場合には、複数のビルディングブロックを一度に投入するだけで対応できます。

また、ビルディングブロック方式は、スモールスタートしてその後拡張を考えるお客様にとっても魅力的なものとなります。スモールスタートし、その後要件に応じて拡張できる「Pay-As-You-Grow」モデルは、初期投資を低く抑えながら、拡張に際しても優れたエクスペリエンスを維持することができます。次のセクションでは、現在利用可能な異なるタイプのインフラストラクチャーのアーキテクチャーと、それぞれがビルディングブロック方式をどのようにサポートするか、あるいはサポートしていないのかについて解説します。

アプリケーションやデスクトップの仮想化、また、広く捉えた場合、EUCソリューションには、現在、3つの基本的なアーキテクチャーが存在します。ビルド・ユア・オウン（BYO）、コンバージド・インフラストラクチャー（CI）、そしてハイパーコンバージド・インフラストラクチャー（HCI）です。

ビルド・ユア・オウン

BYOは、その名が示す通り、アーキテクトやチームが、好みの、あるいは最善の組み合わせと考える製品を独自に選択する方式です。この方式を採用する場合、事前の計画や調査に多くの時間を要し、各製品に関する個別の評価と、それぞれが連携して動作可能かどうかを確認する必要があります。

この方式の場合、構築しようとするソリューションタイプに合わせ、ベンダーが提供するリファレンスアーキテクチャーを選択し、それに従うこともできます。通常このようなリファレンスアーキテクチャーは、特定のベンダーによって提供され、使用する製品はそのベンダーが提供するものが中心となります。DIY方式のリファレンスアーキテクチャーは、時間とリスクの削減に役立ちますが、個別のデザイン要件やユースケース、そしてシステム環境に対して常に適応できるとは限りません。

EUCベースのデザインのためのBYOには、最低でもサーバーとストレージリソースが含まれています。既存ネットワークコネクティビティを利用できるため、ネットワークがBYOのコンポーネントに含まれる必要はありません。図2は、BYOのパーツを簡略化して図示したものです。柔軟な拡張性を持ち、事前にコストを想定することが可能ですが、唯一、ストレージについて問題を抱えています。デザインが許容する最大構成やストレージの選択に依存しますが、複数のストレージアレイやアプライアンスが必要となるのです。ストレージを拡張し、新しいアレイやアプライアンスを追加する場合、その時点で急激にコストが跳ね上がります。

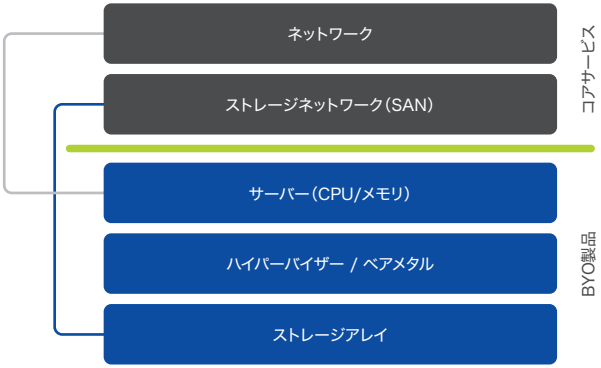

図 2:
ビルド・ユア・オウン(BYO)インフラストラクチャー

過去の経験無しに、同じベンダーあるいは複数のベンダーの製品を組み上げることは、常に大きなリスクを伴います。アーキテクチャーに沿って実際のインフラストラクチャーを購入し、導入が完了するまで、該当ソリューションにおけるパフォーマンスや信頼性は不確実なままです。

このような不確定要素やリスクを受け入れることが可能であれば、BYOは最大の柔軟性を発揮します。一緒に仕事を進めることができるベンダーや製品は、世の中に幾らでも存在し、特定部分については従来のベンダーと推進し、残る部分は新しいベンダーに乗り換えるといった対応もできます。

BYOの場合、サーバーとストレージリソースをそれぞれ独自に拡張することが可能です。拡張性や最大構成は選択した各製品の限界値に依存します。製品は個別に購入されるため、製品を拡張する際の最小単位や構成単位といった内容は、事前には存在しません。つまり、先に述べたビルディングブロックを柔軟に構成できるということです。

コンバージド・インフラストラクチャー

コンバージド・インフラストラクチャーは、2010年前後に市場に登場したアーキテクチャーです。コンバージド・インフラストラクチャーは、通常、BYOで選択した製品をパッケージ化したソリューションです。CIベンダーは、サーバー、ストレージ、ネットワークを製品の中に組み込んでいます。CIのほとんどは、マルチベンダー製品で構成され、単体のソリューションとして提供されますが、CIのすべてのレイヤを自社製品で構成する場合もあります。図3は、コンバージド・インフラストラクチャーの概要を図示したものです。

図 3:
コンバージド・インフラストラクチャー

コンバージド・インフラストラクチャーの場合、馴染み深い製品を単一のソリューションパッケージとして購入することができます。リファレンスアーキテクチャーを具体的な製品として購入する形と同等です。評価するCI製品にも依存しますが、BYO方式で製品を個別に購入した場合以上にリソースを追加できる場合と、そうではない場合があります。

通常、多くのCIベンダーの製品は、インフラストラクチャーを構成するすべてのパーツを単体製品として購入できるようになっています。またCIベンダーは、CIソリューション全体をサポートする統一窓口を提供する必要があります。これは、彼らがソリューションに含まれる全製品をサポートできることを意味します。障害対応のために複数のベンダーとやり取りする必要がなくなるため、お客様にとってのさらなるメリットと言えます。

ほとんどのCIの場合、ソリューションで使用できる製品数には制限が設けられています。CIベンダーは、その制限の範囲内ですべてのパーツや部品が同時に正しく動作することを確認するために事前テストや検証を行うことで、BYO方式で発生するリスクの多くを回避しています。

CI製品が市場で販売されるようになってから数年間が経過していますが、未だ製品を容易に管理できる機能を実装したベンダーは稀です。CIにはBYO方式と同様の製品が含まれているため、同じくバラバラな方法で管理しなければなりません。コンバージド・インフラストラクチャーは、製品やその購入方法を「コンバージ」しますが、ソリューションの運用管理まで統一できている訳ではありません。

コンバージド・インフラストラクチャー製品は、互いに独立してリソースを拡張できる必要があります。つまり、拡張可能な最小単位はあるものの、単純にサーバーを追加することができるということです。ストレージもまたCIで拡張されることの多いリソースですが、その内容は、CIの一部としてどのようなタイプのストレージソリューションを選択するかに大きく依存します。コンバージド・インフラストラクチャー製品には最大構成サイズがあり、サポート可能なサーバー数や内蔵ストレージアレイに応じて、ストレージに関する制限が発生します。

通常、CIの拡張に対する制限値は十分に大きいものですが、CI製品内でリソースを拡張していけば、いつかはその壁に突き当たることになります。そして、それ以上の拡張を望む場合には、CI製品を追加購入する必要があります。デザイン上の最大想定規模にも依存しますが、その時点で、通常の拡張プロセスとは比較にならない非常に大きなコストが発生することになります。

ハイパーコンバージド・インフラストラクチャー

ハイパーコンバージド・アーキテクチャーは、CIの約1年後に市場に登場しました。サーバーリソース、ストレージリソース、そしてマネージメントレイヤを1つのソリューションとして統合した、真のハイパーコンバージド・アーキテクチャーと言えます。ハイパーコンバージド・ソリューションは、BYOやリファレンスアーキテクチャーを使っても実現できますが、真にハイパーコンバージドであるために、製品にはハードウェアアプライアンスが含まれている必要があります。

製品の一部にハードウェアアプライアンスを含めることで、提供ベンダーは、製品に統合されるリソースと共に、インフラストラクチャーのマネージメント機能を含めることができます。図4は、ハイパーコンバージド・インフラストラクチャーの概要を図示したものです。

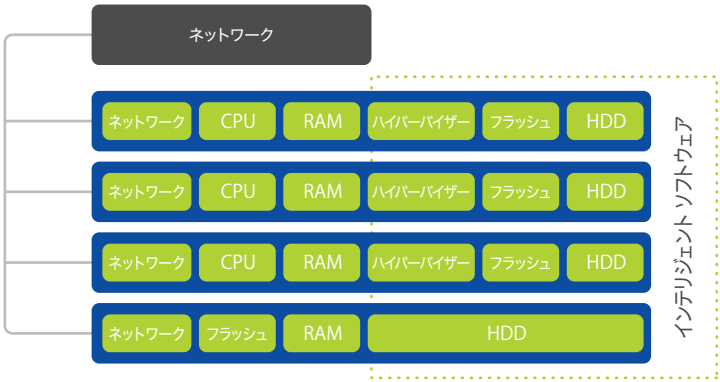

図 4:
ハイパーコンバージド・インフラストラクチャー

真のハイパーコンバージド製品には、他のアーキテクチャーでは提供できない様々な特長があります：

容易なインストレーション – 先進のHCI製品は、高度に自動化されたプロセスを使って、数分あるいは数時間以内でノードをインストールすることが可能です。以前のように、数日～数週間を要することはありません。

容易な拡張性 – 製品を容易に拡張または縮小することが可能です。マネージメントインターフェースを使って、システム環境に新規ノードを容易かつ迅速に追加することができます。

最新のマネージメント – 最新のインターフェースを使って、仮想マシン（VM）単位でマネージメントすることが可能です。システム管理者は、VM単位でパフォーマンスやリソースの消費量、そしてイベントやエラーの発生状況を確認し、容易にVM単位のレポートを作成することができます。

連携機能 – 該当インフラストラクチャーは、ソリューションにおける他のパートとの容易な連携が可能で、かつプログラマティックな制御ができなければなりません。このため、APIやPowerShellコマンドレットいったメソッドを、HCI製品から提供する必要があります。APIを使って製品間の通信や制御を自動化することで、管理や運用を省力化すると共に、システム環境における正確性を向上させます。

パフォーマンスについては、意図的にHCIの特長から除外しました。最新のハイブリッドあるいはフラッシュベースソリューションのパフォーマンスが優れていることは、誰の目から見ても明らかだからです。HCIは、シンプルで効率的なインフラストラクチャー・レイヤを構成することを目指します。HCIを採用することで、IT部門のチームは、余計な調整等に時間を費やすことなく、自動化やアプリケーションといったレベルで、業務部門に対して新しい付加価値を提供することができます。

どんなEUCをデザインする場合でも、ストレージリソースに対する様々な要件が存在します。ストレージには、サーバーベースのVM、ユーザーデータ、仮想デスクトップインフラストラクチャー（VDI、あるいはユーザーVM）などを収容する必要があります。このようなストレージ要件は、システム環境を考える上で最も重要であると同時に、多くのプロジェクトを失敗に陥れたり、問題に苦しみ続ける要因にもなります。

このような理由から、本eBookでのストレージに関する解説は、VDIサービスソリューションに焦点を絞りたいと思います。個々の仮想デスクトップに対する要件は小さく目立つものではありませんが、ストレージが拡張され、大規模に仮想デスクトップをサポートするようになると、要件に適合するよう正しくデザインされていないストレージのパフォーマンスは、瞬く間に劣化します。

妥当なレイテンシの各仮想デスクトップが、それぞれ15 IOPSを要求し、2,000人の同時ユーザーをサポートする規模までの拡張を見込むとすると、これらの合計値は30,000 IOPSになります。この数字は非常に大きいため、平均的なストレージアレイの能力を超えてしまいます。しかし、ストレージソリューションを、単純に環境の平均I/Oに適応するようデザインするだけでは不十分で、デスクトップのブート時やユーザーログイン時のようなピーク時にも耐えられるものでなければなりません。

仮想デスクトップのワークロードは、平均的なエンタープライズデータセンターで稼動する他のワークロードとは非常に異なります。仮想デスクトップのI/O特性がスパイキーな（予告なく急上昇する）ものだからです。例えば、Outlookのようなアプリケーションを最初に起動すると、1ユーザーセッションに対して1,000 IOPSも費やすこともあり得ます。これは先に述べた15 IOPSを遙かに上回る数値です。図5に、他のアプリケーションのIOPの影響度を図示しました。

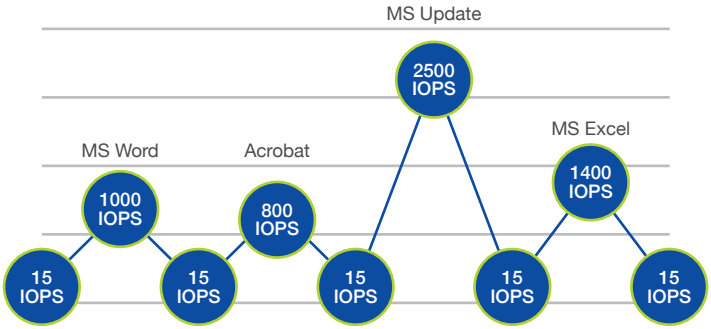

図 5:
VDI IOPS

パッチの適用や環境のリフレッシュといった処理も、非常に大きなIOPSのスパイクを発生させるため、適切な計画を立てた上で臨まなければ、パフォーマンスに大きな影響を与えます。仮想デスクトップを50追加しようとすると、その処理自体が極めて高いI/Oのスパイクを発生させます。従って、メンテナンス操作も考慮に入れて、ストレージアーキテクチャーのピークIOPSを考える必要があります。

フルクローンやシェアド・イメージをVDIソリューションに実装するアーキテクチャーについては様々な方法が存在しますが、キャパシティおよびパフォーマンスにおいてストレージ要件に与える影響は異なります。フルクローンの場合には、キャパシティとストレージの両方について追加が必要となるため、重複排除機能が重要な役割を果たします。また、フルクローンでは、パッチ適用中にI/Oが増加するため、個別にパッチを適用する必要があります。

CitrixのMCSまたはPVS、あるいはVMwareのリンククローンといったシェアド・イメージを採用した場合には、また別のI/O問題が発生します。このようなシェアド・イメージによるアプローチでは、親となるイメージを共有し、各仮想デスクトップ側はそれぞれに固有のデータで必要となるわずかなスペースだけを消費します。シェアド・イメージは、通常のVMと比較して、異なるパフォーマンス要件を必要とします。対象イメージは数百あるいは数千の仮想デスクトップで利用されるため、ブートストームのような状況に対応できる大きな値のIOPSを発生できなければなりません。シェアド・イメージがボトルネックになると、それを利用するすべての仮想デスクトップに悪影響を及ぼし、ユーザーエクスペリエンスが低下します。

このようなピーク値や、様々なアプリケーション、さらにデスクトップ仮想化アーキテクチャーを考慮に入れながら、対象システム環境におけるブートやログイン時のピークと、定常状態のいずれにも耐え得るストレージソリューションをデザインする必要があります。デザインにおけるストレージ要件を理解するためには、既存の物理PC環境のデスクトップに関する調査を実施すべきです（注：本シリーズの最終巻に、この調査プロセスについての説明があります）。このデスクトップ調査では、実際のパフォーマンスやキャパシティをユーザーベースで収集し、サイジングのための算出に役立てます。

事前に想定できない特殊なI/Oを除けば、アプリケーションおよびデスクトップの仮想化におけるストレージ要件について最後に考慮すべきは、デスクトップワークロードではWRITE I/Oが非常に多いという点です。データを読み込み、その結果をユーザーに提供するだけという、典型的なサーバーワークロードとは大きく異なり、通常、仮想デスクトップでは、ディスクに対する書き込み処理に多くの時間を使います。ストレージアレイから見ると、WRITE処理はREAD処理よりも遙かに高負荷なものとなります。典型的なサーバーワークロードでは、READ処理80%に対してWRITE処理は20%程度ですが、定常状態のデスクトップロードでは、その数値が逆転します。ストレージを評価する場合、ブートストーム時に対応できるよう、共通READブロックをキャッシングできるといったストレージが持つ “すばらしい機能” に着目するだけではなく、ストレージソリューションのWRITEに対するバッファリングやコミット方法についても十分に検討する必要があります。

ストレージタイプ

ストレージには様々なタイプがあります。現在利用可能な主要ストレージには、レガシーな階層化ストレージアレイ、ハイブリッドフラシュアレイ、オールフラッシュアレイなどがあります。パフォーマンスやキャパシティをワークロードに提供する方法は、それぞれ異なります。また、各ストレージに対するベンダーの実装方法も異なるため、以下で簡単に解説を行います。

レガシーな階層型アーキテクチャー – これは過去10〜20年間、サーバーベースのワークロードに利用されてきたエンタープライズ向けのレガシーなアレイシステムです。一般的にデュアルコントローラを備えたアーキテクチャーとなっており、ここ10年間で、異なるパフォーマンスやキャパシティを持ったディスクを使って階層構造を形成するよう変化してきました。複数のディスク階層を持つことで、ワークロードによって異なるキャパシティやパフォーマンス要求に応じようというものです。ここでは2つの選択肢が用意されています。1つは、特定のワークロードのためにハイパフォーマンスなディスクを使って、専用のプールを形成するようパフォーマンス重視のデザインを行なう方法です。しかしこの方法は、非常に高価かつ限定的です。もう1つは、このアーキテクチャーに追加された階層化機能を利用し、データのブロックにおける階層の上げ下げを、デマンドに応じてアレイ側に行わせる方法です。この自動階層化における問題は、VDIワークロードに対する階層決定に時間がかかり過ぎることです。

オールフラッシュ – オールフラッシュストレージは、すべてがフラッシュベースのストレージで構成されています。このストレージアレイで使用されるフラッシュには、様々なタイプがあります。最新のオールフラッシュアレイは、フラッシュストレージの特性を活かすようにデザインされており、オペレーティングシステムやファイルシステムについてもフラッシュを考慮に入れたデザインが行われています。中には、レガシーなアレイデザインのままで、単にハードディスクアレイだけをすべてフラッシュに置き換えた製品も存在します。この場合、確かに過去の製品よりは高速ですが、最終製品は、フラッシュの特性に合わせデザインされたものではありません。

オールフラッシュストレージは極めて高速で、比類無きパフォーマンスを提供します。該当アレイがデザインに必要なキャパシティを、適切な価格で提供していることを確認するため、データ重複排除機能やデータ圧縮機能を備えているかどうか見極める必要があります。ほぼすべての最新オールフラッシュアレイは、レガシーなディスクアレイより管理が容易ですが、多くのハイブリッドフラッシュで提供されている容易な管理やVM単位での管理が可能とは限りません。

ハイブリッドフラッシュ – ハイブリッドフラッシュアレイは、フラッシュドライブとハードディスクドライブを組み合わせ、効率的に使用できるようにデザインした最新のアーキテクチャーです。アレイのキャパシティやパフォーマンスをどう引き出すかは、ベンダーのアーキテクチャーによって異なりますが、その結果は類似しています。より小規模なフラッシュで驚異的なパフォーマンスを生み出すと共に、アレイに格納されたハードディスクを使って大容量のデータ保存領域を確保しています。優れたハイブリッドストレージアーキテクチャーでは、フラッシュとハードディスクの間の自動階層化をデマンドベースで行ない、マニュアルチューニングの手間を省き、パフォーマンスに関わる見落としを避けることができます。

最新のVDIデザインに最適となるアーキテクチャーは、ハイブリットおよびオールフラッシュアーキテクチャーです。これらのアーキテクチャーでは、VDI環境に必要となるパフォーマンスを提供できるだけでなく、前述した最新のマネージメントエクスペリエンスも提供することが可能です。本来、VDIワークロードの動きを予測することは非常に困難で、ストレージソリューションがストレージレイヤの選択決定動作やブロックがキャッシングされるのを待つ必要があるとすれば、その完了以前にパフォーマンス要求の方が消滅してしまい、エクスペリエンスに悪影響を及ぼしています。

サーバーのサイジング

デザインにおけるサーバーレイヤのサイジングついては様々な考え方があります。まず “スケールアップ” ですが、これは、より少ない数の大規模なホストを使ってリソースを提供しょうというアプローチであり、“スケールアウト”は、より多くの小規模なホストを使ってリソースを提供しようというアプローチです。2ソケットのホストを使用し、デザイン上のCPU集約率 (Consolidation Ratio) 制限に反することなく最大の集約率が得られる、両者の中間に位置するようなアプローチが最も望ましい方法と言えます。本eBookでは、VDIワークロードに焦点を当てたサーバーリソースのサイジングについて説明します。

サーバーリソースのサイジングを行なう場合、注意すべき重要な計算値が3つあります。それが、各ホストの物理メモリサイズ、CPUクロック速度の合計、CPUのコア数とvCPU比率です。最初に、VDIをデザインする際にはメモリサイズをオーバーコミットすべきではありません。この警告を無視した場合、得られるものが何も無いだけでなく、システム環境においてパフォーマンスの問題を引き起こす可能性があります。

CPUクロック速度の計算は、先に述べたデスクトップアセスメントの結果に大きく依存します。アセスメントの内容から、ユーザーセッションが平均、あるいはピーク時に使用しているCPUについて理解できるからです。また、このアセスメント結果から、使用されているメモリ量の詳細についても算出できます。

ホストや仮想化クラスタに関する他の推奨内容としては、ホストの使用率を80%以上にしないことと、クラスタを常にN+1でサイジングすることがあげられます。80%に収まるホスト使用率は、アプリケーションやデスクトップの仮想化に限った話ではなく、ハイパーバイザーで稼動するすべてのワークロードに対しても同様に重要となります。ホストの使用率が80%を超えた場合、ピーク時における余裕がなくなります。クラスタサイズにもよりますが、ホストに障害が発生した場合には、十分なリソースの提供が困難になります。また、クラスタをN+1でサイジングするという対応は、ホストが単体障害を起こした場合でも、クラスタに十分なリソースを残し、すべてのVMの稼動を確保し、障害が発生したホストを問題なく再起動できるようにするためのものです。多くのお客様は、ホストの単体障害に対する回復力確保を求め、より厳しいSLA要件を持つ一部のお客様はN+2の対応を必要とします。

サーバーサイジングに関する最後の話題は、仮想CPU数と物理CPU数に注目したCPU比率（vCPU:pCPU）です。この比率を高く設定し過ぎると、CPUのスケジューリングに問題が発生し、パフォーマンスやユーザーエクスペリエンスに重大な影響を与えるため、非常に重要な比率と言えます。たとえば、vSphereホストでスケジューリング上の問題が発生すると、CPU待機時間が増加し、スケジューラーに障害が発生したとみなされ、vCPUでスケジュールされたもの全てがpCPUに移されます。つまり、リソースに空きが存在するにも関わらず、vCPUで待ち時間が発生することになります。CPU比率は、VMwareクラスタ上で仮想化されたワークロードのタイプに応じて異なります。通常、サーバーやデータベースに対する比率は小さく、VDIワークロードは高めになります。

vCPUの使用は線形的に計算できるものではありません。つまり、すべてのVMに1つずつvCPUを割り当てるようにすれば、より集約率の高いホストを構築することができるということです。多くのVMに2つ以上のvCPUが割り当てられている場合、計算上で影響が発生します。vCPUを2倍割り当てようとする際に、単純に2で割ればよいというものではありません。図6は、実際に顧客先で稼動する範囲を示したものです。CPUの製造業者が実施する意図的なテストでは、もっと高い比率が示されるかも知れません。しかし、このような数値は、現実世界のデザインに適用できるとは限らないため、注意することが必要です。

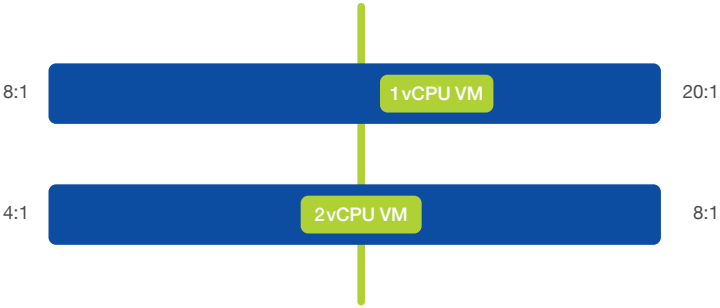

図 6:
VDI集約率はvCPUとvCPUの比率に大きく左右されます。計算は、設定する仮想デスクトップのvCPU数によって異なります。この図は、安全であることが実証された経験値の範囲を示したものです。

単一vCPUの仮想デスクトップが正常に動作する範囲は、8：1から20：1の間となります。該当範囲は広く、どの値となるかは選択内容によって異なります。1つは、ホストの大きさとホストあたりのVM数であり、お客様が快適だと感じるレベルの数値を示しています。例えば、18コアCPU2基から構成されるデュアルソケットホストなどが該当します。これで最大700以上のVMに対して、適切なメモリ容量と十分なクロック速度を提供することができます。通常お客様は、このように多くのVMを単一のホスト上で稼動させることを嫌います。そこで2つのシナリオが考えられます。1つ目は、誰かが意図的に設定した限界値に従って密度を下げることです。比率における最低値を選択した場合、同じホスト上のVM数は288となります。2つ目の選択肢は、少ないコア数をもったCPUを選択すると共に、比率を中間値に設定することです。

もし、12コアCPUで 12：1 の比率を選択した場合、VM数は288となります。この比率は、通常、お客様からのフィードバックやアーキテクトの推奨値、そしてインフラストラクチャーのコストなどによって決定されます。異なるCPU構成を選択することで、大幅なコストセーブを実現できる可能性もあります。

デュアルvCPU仮想デスクトップの場合の計算も同様ですが、唯一異なるのは、vCPUの合計が2倍になることです。運用範囲は、4:1 から 8:1 の間になります。もっと高い数値を提示するベンダーもありますが、推奨値は実際のお客様の状況に応じて変化します。異なるCPU比率範囲を用い、前述の例と同様な方法でで決定します。

許容範囲の中間でCPU比率を選択した場合には、集積密度を上げる余地が生まれますが、重要となるのはシステム環境が許容範囲で動作し続けられるようにすることです。また現状は、このようにCPU比率を設定できるツールが存在しないことにも注意を払う必要があります。デザイン時に一旦決定された特性値に従い、その後システム環境の管理や拡張を行なう必要があります。メモリやクロック速度と同様、クラスタにVMを追加したり、ホストを追加してリソースを増やすことができる余地を考慮に入れ、CPU比率を計算および決定しなければなりません。

CPU比率については、データを収集して手計算することで算出が可能です。PowerShellスクリプトを使ってデータを収集し、比率を計算させているシステム管理者もいます。スクリプトを使い、日々それを定期的なジョブとして流すことで、比率を逸脱していないか、あるいはクラスタが危険な状態にないかどうかを確認することができます。

RAMやメモリバスの動作周波数もまた、サーバーのサイジングに関係します。メモリをサイジングする場合には、予算が許す範囲で最高速度のバススピードを持った最高密度のメモリを採用することが得策です。メモリの低速性が原因でRAMに対するREAD/WRITE処理の完了待ちが発生し、CPUサイクルにアイドル状態が生まれるといった問題がしばしば発生します。

EUCのデザインにおいて、複数の仮想化クラスタを構築する背景には、いくつかの理由があります。通常は、稼動させるワークロードやクラスタサイズに応じて、複数のクラスタを導入すべきかどうかを決定します。本eBookでは、この話題について深く掘り下げる余裕がありませんが、他の解説書やオンライン上でも話題に取り上げられている幾つかの推奨事項があります。

まず重要なのは、数百以上のユーザーを対象とするVDIをデザインする場合、VDIワークロードのためのインフラストラクチャーと、仮想化をマネージメントするためのインフラストラクチャーを分離することです。つまり、マネージメントサーバー、VDIブローカー、ファイルサーバー、アプリケーションマネージメントサーバー他、仮想デスクトップ以外の機能は全て別のクラスタ上で動かすことです。該当EUCデザインにおいて、専用のマネージメントクラスタが必要か否かは、システム環境の規模に依存します。小規模なデザインであれば、マネージメントVMを既存のサーバー仮想化クラスタ上で動かすことも可能です。

仮想デスクトップクラスタは、16〜32ホストまで拡張することが可能です。この範囲であれば、VMが必要とする十分に大きなリソースプールを形成することが可能で、一般的なサイズより大きいクラスタをお客様に提供することができます。最近のハイパーバイザーでは、クラスタとして64ホストまでを許容するようになっていますが、ほとんどのアーキテクトやお客様の場合、まだそこまでの規模は不要でしょう。仮に、このような範囲を超えるような大規模なシステム環境となる場合には、VDIクラスタを複数用意する必要があります。

システム環境規模に加え、複数の仮想化クラスタを用意する理由は、異なる特性を持つワークロードに対応するためです。VDIクラスタ上には様々なワークロードが混在しています。大規模な数の1 vCPUあるいは2 vCPUの仮想デスクトップが存在する場合、それぞれに対して別のクラスタを立てるようデザインすべきです。これによって、それぞれのクラスタで異なるCPU比率に対応でき、デザイン自体も容易になります。異なるCPU構成を混在させた場合には、新たに合成比率を計算することが必要になり混乱が発生します。

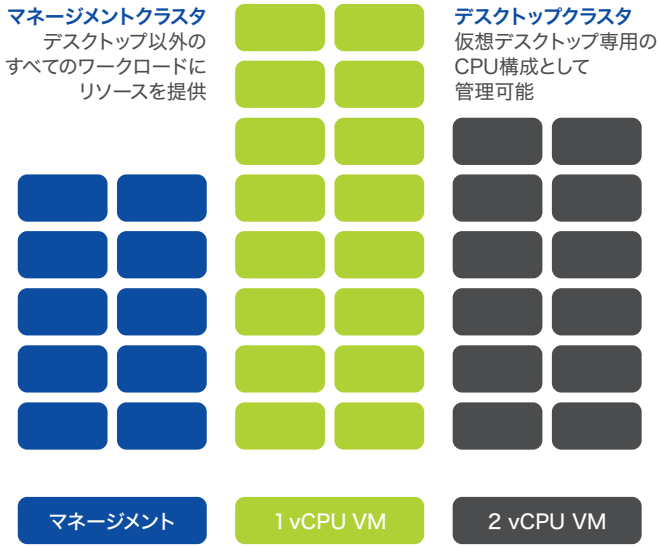

図 7:
マネージメントおよびデスクトップクラスタ

ここまでご覧いただくことで、デスクトップやアプリケーションにおける仮想化の可能性について興味が湧いてきたと思います。もし皆様や皆様の会社が、どうすれば間違いなくデスクトップやアプリケーションの仮想化を実現できるか理解したいとお考えなら、Nutanixがお役に立ちます。既に数々のアワードを受賞しているNutanixのWebスケールなアーキテクチャーは、その存在を意識させない「インビジブル」なインフラストラクチャーとして、もっとも優れたVDIプラットフォームの容易な構築を実現します。

スタートするにあたって

Nutanixでは、デスクトップやアプリケーションの仮想化を確実に成功させる最善の方法をご提示できるように幅広く、そして深い考察を加えます。

現在のお客様の環境の理解：

該当プロセスは、お客様の現在のエンドユーザー環境を完全に理解することから始まります。例えば：

- **エンドユーザー指標**：使用しているアプリケーション、エンドユーザーがアクセスするデバイス、使用場所やコネクティビティなど、エンドユーザーのプロファイルや関連する要素を収集します。
- **ネットワークサービスおよびインフラストラクチャーに固有な指標**：ファイルサービス、認証やアクセスコントロール、ファイアウォール、ロードバランシングなど、様々なエンドユーザーサービスに関する適切な情報を収集します。また、パフォーマンス、レイテンシ、スループットといった情報も、同時に収集します。
- **すべての情報を開示**：説明責任を果たすことが重要な成功要因です。

新しい環境のサイジング：

上記の情報を活用することで、新しい環境を正確にサイジングすることができます。Nutanix Sizerを使って直接サイジングできることに加え、以下のようなガイドラインも提供します：

- 主要なサーバーやデスクトップの高可用性を常に考慮します。

- 以下を考慮に入れ、追加でインフラストラクチャーやクラスタが必要かどうかを判断します：
  - **業務面：**SLA、ライセンス、セキュリティ、予算、方針
  - **移行計画：**Nutanixや業界のベストプラクティス、P2V移行ガイドラインに従うと共に、デスクトップの生成で使用するゴールデンイメージの作成に周到な注意を払います。既存環境の移行であれば、Nutanixのパートナー各社をご利用いただくか、また可能であれば、ネイティブなツールの使用を推奨します。

Nutanixグローバルサービスは、インフラストラクチャー構築の成功に向け、これらの手順の全工程、または一部についてもご支援します。Nutanixでは、グローバルサービス部門を通じて、デスクトップ仮想化プロジェクトでインフラストラクチャーのサイジングを誤ってしまうというリスクを回避できる業界唯一のソリューションを提供します。

さらに、NutanixのVDI保証制度により、エンドユーザーVDIに必要なサーバー（仮想CPUやメモリ）およびストレージ（パフォーマンスとキャパシティ）リソースを確実に提供することができます。システム環境におけるVDIユーザー数とタイプを決定するだけで、NutanixのVDI保証制度によって、インフラストラクチャーのサイジングに関わるリスクを排除することができます。

デスクトップやアプリケーションのための「インビジブル・インフラストラクチャー」に関する詳細については www.nutanix.jp/vdi をご覧いただくか、team-japan@nutanix.com宛てに直接ご連絡ください。また、Twitterは DM@nutanixでフォローいただけます。自社のデスクトップやアプリケーションの仮想化環境において、豊富な実績と様々な認定を持つNutanixのソリューションを最大限活用するための説明やデモを直接ご覧になりたいお客様は、http://www.nutanix.jp/demo/からリクエストいただくことが可能です。

Nutanixの検討はまだ早いとお考えですか？ Nutanix Nextオンラインコミュニティ（next.nutanix.com）を通じて、Nutanixのエキスパートやお客様と繋がることができます。

Nutanixは、ITインフラストラクチャーをその存在さえ意識させない「インビジブル」なものに変革することで、企業のIT部門が、ビジネスに直結したアプリケーションやサービスの提供に注力できるようにします。Nutanixのエンタープライズ向けクラウドプラットフォームは、オンプレミスのインフラストラクチャーの特性である優れた予測性能やセキュリティそして管理機能と共に、パブリッククラウドが持つ俊敏性と経済性そしてシンプルな運用性能を提供します。Nutanixの提供ソリューションは、Webスケール技術とコンシューマーグレードなデザインによって、サーバー、仮想化機能、そしてストレージを、耐障害性能に優れたソフトウェア・デファインドなソリューションとして統合することで、あらゆるアプリケーションをどのような規模でも稼動させることができます。

詳細については www.nutanix.jp をご覧ください
Twitterは@nutanixでフォローいただけます

NUTANIX™

www.nutanix.jp
Email team-japan@nutanix.com