三重大学教育学部附属教育実践総合センター紀要　2014,　第 34 号，13 － 18 頁

ネットワーク演習ための仮想サーバ構築

丁　亜希*・山守　一徳*

ネットワーク関連の演習では、学習者の一人一人に 1 台の UNIX サーバマシン及び 1 台の Windows マシンを与えて、色々な課題を完成させることがある。そのためには Hyper-V など仮想化ツールを用いて仮想的な UNIX サーバを構築し、学習者に提供することが考えられる。Hyper-V を用いて仮想サーバを構築するにあたり、Hyper-V マネージャーという GUI 方式の管理用ツールを利用して手作業で構築できるが、必要な仮想サーバ数の多さや演習中の仮想サーバ再構築の必要性から考えると、これは演習指導者にとって大きな負担になる。

そこで、本研究では、コマンドレットを用いて仮想サーバの自動構築方法を考案した。この方法では、予め用意されたテンプレートから PowerShell スクリプトによって自動的に演習に必要な台数分の仮想サーバを複製し、複製された仮想サーバが各自の再設定を自動的に行う。これで演習指導者は最初のテンプレートだけを手作業で作成しておけば済む。

キーワード：ネットワーク演習　仮想サーバ　Hyper-V　PowerShell

1．はじめに

コンピュータ・ネットワーク関連の授業では、一人の学習者に 1 台の Windows マシンと、もう 1 台の UNIX サーバマシンを与え、いろいろな演習課題を完成させることがある。

このような場合は、数多くのコンピュータハードウェアが必要になるだけでなく、演習中に学習者に与える各課題において当該課題に適したサーバ環境を提供しなければならないので、同一のハードウェア上に何回もサーバシステムの再設定や再構築が必要になる。また、学習者の操作ミス等によってサーバマシンが使用不能になることがあり、その場合には各課題開始時点のサーバマシン環境に迅速に回復し演習を再開させる必要がある。コンピュータハードウェアの数を減らし、より便利にサーバマシンを作成または回復するためには、仮想化ツールを用いて学習者に仮想サーバ環境を提供することが有効であると考えられる[1)]。

現在、数多くの仮想化ツールが開発され、様々な目的や用途に合わせて運用されている。ハイパーバイザ型の仮想化ツールには Hyper-V，ESXi，XenServer，KVM などがある。また、ホスト型の仮想化ツールには Virtual PC，VMware Player，VirtualBox などがある。仮想サーバ環境を構築する際に、ネットワーク関連授業と演習の視点から、どの仮想化ツールを選ぶと良いのか、その選択によって構築された仮想サーバの性能、コストおよび管理手間などはどうなっているのかが問題である。これらの問題に答えるには、より多くの実践的な教育・研究が必要であろうが、ここでは、本教育課程の持つライセンス事情により、仮想サーバ環境の構築に Hyper-V[2)] を用いることにする。

Hyper-V には、仮想マシンの作成や管理などのためのユーザインタフェース「Hyper-V マネージャー」がある。これを使うと、新規仮想マシンや仮想ハードディスクなどの作成は、Hyper-V マネージャーのメニューから対話的な方式のウィザードを起動して容易に行うことができる。しかしながら、このような作業は簡単ではあるが端末の前で手作業で行わなければならない。また、作成された仮想 UNIX マシンの物理アドレスや IP アドレスなどの基本設定も個別に UNIX マシンにログインして手作業で行う必要がある。数多くの仮想サーバの作成や繰返しての仮想サーバの作成などが必要な場合には、この方法では、向かないと言えよう。

一方、Hyper-V には、PowerShell[3)] で利用できる 170 個ほどの管理用コマンドレットが用意されている。これらのコマンドレットを用いる PowerShell のスクリプトを作成したら、仮想サーバ作成の自動化を図ることができると考えられる。

本研究では、本学部でのネットワーク関連授業の演習において、仮想サーバ環境の自動構築方法を考案する。この方法では、まず手作業でテンプレートとなる仮想サーバ CentOS 6.4 を作成して、必要な設定を完成する。そして、PowerShell スクリプトによって、テンプレートから複数台の仮想サーバを自動的に複製する。同時に、スクリプトは CentOS 6.4 の基本設定に必要な物理アドレス、IP アドレスおよびホスト名などの情報を集め、複製された個々の仮想サーバに渡す。これらの仮想サーバは渡された情報に従って各自のネットワーク環境の再

*三重大学教育学部

設定を行う。この方法によって、演習指導者は最初のテンプレートだけを手作業で作成しておけば済む。

2．ネットワーク関連演習と仮想環境構築

情報教育課程のネットワーク関連演習は、いくつかの授業の中に含まれている。それらの演習は、ネットワークシステムの運用や管理に必要な基礎知識と技術を習得することを目標とし、サーバ用 OS（現在では主に CentOS 6.4 を使っている）の新規インストールからよく使われる応用サーバの設定や管理方法までの演習、または、通信ソフトウェア開発などの演習を含む。学習者が Windows マシンから Telnet などを使ってサーバマシンにログインして、遠隔操作のコマンド方式で各種の設定・管理操作や通信プログラムの開発を行う。

学習内容は、OS インストール、ユーザ管理、SSH 設定と管理、メールサーバ設定と管理、メーリングリスト管理、DNS サーバ設定と管理、WEB サーバ設定と管理、ファイアウォール設定と管理、侵入検知、プログラム開発などの多種多様であり、それらに合わせてさまざまな課題を設計しなければならない。個々の学習内容に対応する演習課題の設計や、演習効果の確認や評価などについては、別のチャンスで詳しく論じたいが、1 つの仮想サーバを用いていくつかの課題を継続的に実施させる場合もあるし、課題ごとにそれぞれ専用の仮想サーバが必要となる場合もあることが明らかであろう。仮想環境を構築する際に、学習者数だけでなく、必要な仮想サーバの種類も多く必要であることも意識しなければならない。

演習用仮想サーバ環境を構築する方法として、分散型と集中型との 2 種類が考えられる。分散型とは、個々の学習者用の Windows マシンにそれぞれ Hyper-V を導入し、その上に学習者個人用の仮想サーバを作成する方法である。集中型とは、学習者用マシンと別に、1 台または数台のサーバ専用マシンを用意し、その上に複数台の仮想サーバを作成し、個々の学習者に提供する方法である。

分散型では、各学習者が自ら Hyper-V マネージャーを用いて仮想サーバの起動・終了・回復・入替などの管理操作をすることが可能になり、演習指導者の負担を減らすことだけでなく、学習者にとって Hyper-V の操作方法も勉強することになる。また、マシンの負荷が分散することにより快適な演習環境を作ることが期待できる。一方、Hyper-V を導入できるシステム要件として、OS は Windows 8 Pro 以上であることと、CPU は SLAT 対応のことであるので、ある程度のスペックのハードウェアが要求される。特に、ゼロクライアントのような情報端末教室やネットワーク関連演習以外の学習と共用される環境では、学習者用マシンに Hyper-V を導入することがあまり期待できない。

集中型では、仮想サーバの管理操作は指導者がサーバ専用マシン上で行い、指導者の負担が重くなるかもしれないが、集中管理なので Hyper-V 管理コマンドレットによるスクリプトなどを活用すれば、分散型より管理操作が楽になることも考えられる。また、学習者にとって、仮想サーバ管理操作に関わりがなく、与えられた課題に専念することが期待できる。学習者に使われるコンピュータは比較的低スペックのものでもよい。しかしサーバ専用マシンに負荷が集中することによって、快適な学習環境が得られない恐れがあり、事前に十分な実験や調査が必要であろう。

3．集中型仮想サーバ環境の構築方法

Hyper-V の利用条件や仮想サーバ構築方法などについて、文献 1 を参照されたいが、ここでは、集中型の仮想サーバ環境の構築方法を考案する。つまり、1 台だけの Windows マシンに Hyper-V を導入し、その上で演習に必要数台分の仮想マシンを作り、さらにそれらの仮想マシンの上で CentOS 6.4 を稼働させる環境を構築することを考える。以降、Hyper-V を導入した Windows マシンのことを管理ホストと呼び、CentOS 6.4 などがインストールされている仮想マシンを仮想サーバと呼ぶことにする。

以下、複数の課題ではなく、1 つの課題だけを考える場合の仮想サーバ環境の構築手順を述べる。

手順 1）管理ホストの上で、1 台目の仮想マシンを作成する。
手順 2）仮想マシンを起動し、その上で課題の目的と内容に合わせて OS のインストールやその他の必要な設定を行う。設定完了の仮想サーバ（テンプレート）をシャットダウンする。
手順 3）管理ホストの上で、テンプレートから新しい仮想サーバを複製する。
手順 4）新しい仮想サーバを起動し、その上で必要な再設定を行う。設定完了後、シャットダウンする。
手順 5）新しい仮想サーバのネットワークアダプタに新たに MAC アドレスを割り付ける。
手順 6）学習者数に合せて必要な台数分まで、手順 3）～手順 5）を繰り返す。

手順 1）では、仮想マシンに必要な主記憶、ハードディスク、ネットワークアダプタの設定と作成を行う。ここで、特に注意が必要なのは、ネットワークアダプタに固定物理アドレス（MAC アドレス）を割り付けることである。デフォルトとして、Hyper-V は一定の範囲内で

動的に各仮想マシンに MAC アドレスを割り付ける。一方、CentOS は、MAC アドレスが変更されたら、新しいネットワークアダプタが追加または入れ替えられたと認識する。動的に割り当てられる場合は、設定済のネットワークインタフェース（例えば eth0）の他に、新たにネットワークインタフェース（eth1, eth2 など）を自動的に作成することがある。こうなると、設定済のネットワークインタフェースが機能しなくなる。

手順 2）では、設定項目が各課題内容に従って異なる。OS インストールの課題ならば、特に何も設定しなくても良い。特定のアプリケーションを利用する課題の場合は、当該アプリケーションに合せて、ソフトウェアの追加インストールや環境設定が必要である。手間の掛かる作業になることもある。

従って、個々の仮想サーバを新規に作成することより、設定済のテンプレートから新しい仮想サーバを複製することの方が賢明である。手順 3）では、そのような複製の作業を行う。

しかし、単純に複製だけ行えば、すべての仮想サーバでは、ホスト名やアドレスなどの設定を含めて同じ環境になる。手順 4）では、複製された各仮想サーバに対して必要な再設定を行う。再設定を行うためには、仮想サーバを起動する必要があるが、同じホスト名やアドレスを持つ仮想サーバを複数同時に稼働させることができない。従って、このような作業は逐次に行わなければならない。

最後に、手順 5）で固定 MAC アドレスを複製された仮想サーバに割り付ける。割り付けた後も、仮想サーバのネットワーク設定も相応に変更しなければならないが、その変更は手順 4）で済ませばよい。

4．仮想サーバ環境構築の自動化

前述の仮想サーバ環境構築方法の中で、テンプレートを作成する手順 1）と手順 2）は 1 回だけの作業なので、Hyper-V マネージャーの対話方式や仮想サーバにログインしてコマンドを実行させるような手作業で行っても構わない。しかし、手順 3）～手順 5）は、何回も繰り返す作業なので、手作業の代わりに複製作業の自動化を考える必要である。

仮想サーバを複製する方法の一つとしては、まずテンプレートの仮想ハードディスク（vhdx）をコピーし、それからこのコピーをベースにして下記のいくつかのコマンドレットを使って新しい仮想マシンを複製すればよい。

New-VM　指定された vhdx から仮想マシンを作成する
Set-VMMemory　主記憶容量などを設定する
Add-VMNetworkAdapter　ネットワークアダプタを加える
Set-VMNetworkAdapter　ネットワークアダプタに MAC アドレス等を設定する

コマンドレットの詳細は、文献 4 を参照されたいが、これらのコマンドレットを管理ホスト上の PowerShell スクリプトに入れて必要な台数分まで繰り返し実行させると複製の自動化が実現できる。

手順 4）の複製後の CentOS 6.4 の再設定自動化を考える。複製後の再設定を 1 回かつ 1 回のみ実行させるためには、手順 2）のテンプレートをシャットダウンする直前に touch /etc/reconfig 命令で、空のファイルを作成し、さらに/etc/rc.d/rc.local の最後に次の数行を追加する：

```
if [ -e /etc/reconfig ]; then
    /bin/rm /etc/reconfig
    再設定の処理の記述
    poweroff
fi
```

つまり、/etc/reconfig というファイルが存在すれば、それを削除して、再設定の処理を行ってから、テンプレートをシャットダウンする。

再設定の処理の内容は演習課題の内容によって異なる。複製された仮想サーバと複製元のテンプレートとの間で必ず違うのは、ホスト名、IP アドレスおよび MAC アドレスだけであるため、ここではこの 3 つの項目の再設定だけを考えることにする。電子メールや Web サーバなどのアプリケーションが必要となる課題の場合は、当該アプリケーションの固有な設定ファイルを書き換える処理も必要であるが、再設定された新しいホスト名と IP アドレスに従って必要な書き換え処理を上記の if 文の「再設定の処理の記述」のところに追加すればよい。

CentOS 6.4 では、ホスト名、IP アドレスおよび MAC アドレスに関連する設定ファイルは以下の 3 つである。

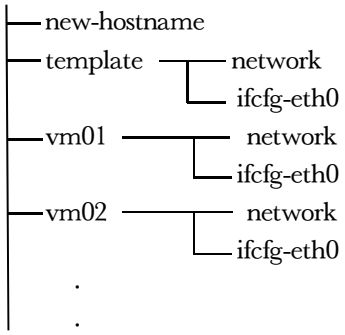
/etc/network
/etc/sysconfig/network-scripts/ifcfg-eth0
/etc/udev/rules.d/70-persistent-net.rules

70-persistent-net.rules は、もし存在しなければシステム初期化の時に自動的に生成される。手順 5）で新たな MAC アドレスが割り付けられ、仮想サーバが再起動時にその MAC アドレスに合せて 70-persistent-net.rules を再生成するから、ここで再設定処理として、単にこのファイルを削除すればよい。network と ifcfg-eth0 の再設定処理に必要なデータは管理ホストから仮想サーバに引き渡せば良いが、残念ながら、現段階の Hyper-V ではそのような手段が提供されてない。因みに、仮想マシンにインストールされている OS は Windows 系のものならば、Copy-VMFile コマンドレットで管理ホストから仮

想マシンへのデータの引き渡しが可能であるが、UNIX 系 OS の場合はまだできないようである。

そこで、管理ホストと仮想サーバとの間でのデータ引き渡しのために、1 つの共通仮想ハードディスクを作ることにする。管理ホストの Windows 上で数十〜数百メガバイトの大きさの共通仮想ハードディスク comm.vhdx を作成し、NTFS 形式でフォーマットしておけば、Windows からも CentOS からもマウントして利用することができる。

データの引き渡し方法がこれでできたので、次はどのようなデータを渡して再設定を行うかを考える。MAC アドレスなどのデータを管理ホストから仮想サーバに渡し、仮想サーバ側でシェル・スクリプトを作成して network ファイルと ifcfg-eth0 ファイルを書き換えてもよい。しかし、CentOS のシェルより PowerShell の方がもっと強力なので、ここでは、書き換え処理を管理ホスト側で行い、書き換え処理が完成したファイルを CentOS に渡すという方法を採る。そのために、共通仮想ハードディスクを次のような形で構成する。



ここで、template フォルダの下にあるのは手順 2）で設定した network ファイルと ifcfg-eth0 ファイルのコピーであり、テンプレートの設定時に用意しておく。vm01, vm02…フォルダの下には複製後の各仮想マシンのために用意した再設定用ファイルであり、管理ホストで実行される PowerShell スクリプトによって生成される。また、new-hostname ファイルには次に再設定を行う対象の仮想マシンのフォルダ名が入っている。そうしておくと、仮想マシン側での再設定の処理は次のように行う（付録 1 を参照）：

共通仮想ハードディスクをマウントする
new-hostname からフォルダ名を取得する
当該フォルダ下のファイルをコピーする
共通仮想ハードディスクをアンマウントする

ただ、管理ホストで作られたファイルの改行コードは CR+LF なので、余分の CR を取り除くために cp コマンドの代わりに tr コマンドを使う。

一方、管理ホスト側での PowerShell スクリプト内の主な処理流れをまとめると、次のようになる（付録 2 を参照）：

仮想マシンを複製する
共通仮想ハードディスクをマウントする
フォルダ template からファイルを取り出す
書き換えを行い、結果を対応フォルダに格納する
対応フォルダ名を new-hostname に格納する
共通仮想ハードディスクをアンマウントする
共通仮想ハードディスクを仮想マシンにつける
仮想マシンを起動する
仮想マシンの再設定処理完了まで待つ
共通仮想ハードディスクを仮想マシンから外す
仮想マシンに MAC アドレスを割り付ける

以上を必要な台数分まで繰り返す。

共通仮想ハードディスクをマウントしてからファイルにアクセスする時には、管理ホスト上でどのドライブ文字がそれに割り当てられているかを調べる必要がある。そのために、

Mount-VHD，Get-Disk，Get-Partition

3 つのコマンドレットをパイプラインで連結して、得られたオブジェクトから、ドライブ文字を表す DriveLetter プロパティを取得する。

最後に、PowerShell スクリプト引数を簡略に説明する。

-fromVM　テンプレートの指定、必須
-startIPaddress　仮想サーバの開始 IP アドレス、必須
-toPath　複製先フォルダの指定
-commVHD　共通仮想ハードディスクの指定
-prefix　複製される仮想マシン名の前半の指定
-suffix　複製される仮想マシン名の後半の指定
-number　複製台数の指定

複製台数が指定されなかった場合は、1 台だけとする。仮想マシン名は、前半と後半を連結する形で構成される。デフォルトでは、前半は vm とする。後半は 1 から始め、複数台の複製の場合は順に 1 ずつ増やしていく。従って、前半・後半の指定はなかった場合は、vm01, vm02…のように仮想マシンを複製することになる。また、仮想サーバのホスト名は、vm01.edu.mie-u.ac.jp のように自動的に生成する。仮想サーバの IP アドレスも、指定されたアドレスから順に 1 ずつ増やして割り付けることにする。なお、MAC アドレスは現在すべての仮想マシンに使われている MAC アドレスの最大値＋1 から開始して割り付ける。

5．実験と考察

情報教育課程の「コンピュータ・ネットワーク」授業のために仮想サーバ環境を作ってみた。この授業は通信ソフトウェア開発技法の習得を目的とし、UNIX 上の

ソケット API を使う C 言語プログラムや Windows 上の .NET クラスライブラリを使う C＃言語プログラムの作成する演習も含まれている。受講者は Windows から Telnet で各自の仮想サーバにログインして演習を行う。なお、今年の受講者数が 16 名であり、それほど多くないので、仮想サーバ環境を集中型とし、1 台だけの管理ホストを使うことにする。この管理ホストでは、CPU が i7-3930 K、主記憶容量が 16 GB、OS が Windows 8.1 である。テンプレートとしての仮想サーバ ie00 は CentOS 6.4 で、開発環境のインストールおよび受講者用アカウントや sshd などの設定が完了状態になっている。また、管理ホスト上で PowerShell スクリプトを実行するために、まずデフォルトの実行セキュリティ・ポリシー Restricted を変更する必要がある。Windows 管理者として PowerShell を起動し

```
Set-ExecutionPolicy  RemoteSigned
```

を実行すれば、ローカルディスクに保存されているスクリプトが実行可能になる。以上の準備ができたら、下記のように

```
./Copy-VM.ps1 -fromVM ie00 -startIPaddress 133.67.84.21
 -suffix 21 -number 16
```

スクリプトを実行すると、vm21, vm22, …, vm36 の 16 台仮想サーバが自動的に出来上がる。1 台の仮想サーバ作成にあたり、約 2.5 分かかる。それは、テンプレート仮想ハードディスクの大きさが 5 GB ほどであるから、主にそれのコピーに時間を費やしたと思われる。

Get-VM | Start-VM を実行すると、テンプレートを含む 17 台の仮想サーバが起動される。ここで、起動前後のシステム全体のパフォーマンスを調べるために、Get-Counter コマンドレットを利用して、10 秒ごとに CPU 利用率、主記憶可利用量、ハードディスクやネットワークのデータ転送量を計測した。図 1 に示されるように、結果として、起動される仮想サーバ数が、管理ホストの CPU i7-3930 K の仮想プロセッサ数 12 を超えているにもかかわらず、CPU の最大利用率は 52%を超えることがなく、起動終了後はその利用率は数%のレベルを維持していることが分かった。一方、管理ホストの主記憶使用量は恒定的に総容量の約 65%となり、特に問題にならないようである。

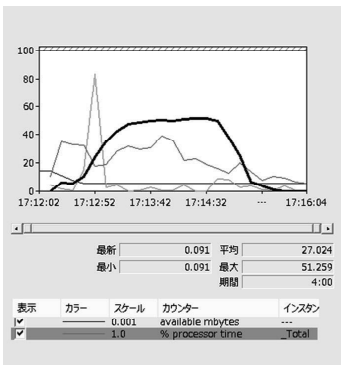


図 1. 仮想サーバ起動時負荷測定結果

「コンピュータ・ネットワーク」授業の中でも、同じようにシステムパフォーマンスを計測した。今回の演習内容は xinetd によって起動されるカレンダーを表示するサーバの設定、およびクライアントプログラム作成である。サーバ設定には、Windows から各自の仮想サーバにログインして、xinetd のインストールと設定から、ファイアウォールのポート解放までの一連の作業を行う。そのために学習者一人ずつ 1 台の仮想サーバを与える必要があった。また、クライアントプログラムの作成には、man での関数などのマニュアル調べ、vi でのプログラム編集や gcc でのコンパイルなどの作業を行う。演習中、以上すべての作業は、重く感じることもなくスムーズに進行されていたことが確認できた。また、管理ホスト上でのパフォーマンス計測データを観察しても、ピーク時に CPU 利用率が 20%を超えることがなく、システム全体の負荷がそれほど高くないことが確認できた。

6．おわりに

ネットワーク関連演習のために仮想サーバ環境構築の自動化方法を考案した。それによって、演習指導者が各演習課題のためのテンプレート仮想サーバだけを作成すれば済む。演習の事前準備段階での負担を減らしただけでなく、演習中で操作ミスなどによる仮想サーバの再構築が必要な場合も 1 つの命令だけで回復させることができるようになった。

また、集中型の仮想サーバ環境のパフォーマンスを考察した。受講者 20 名以下の小規模授業やプログラミングのようなそれほど負荷のない演習の場合、1 台の管理ホストを用いて、各仮想サーバに 1 つだけの仮想プロセッサおよび 512 MB 主記憶だけを割り当ても対応できることを確認した。ただ、その他の内容的に重い作業が必要となる演習や受講者数がより多い場合においては、今後さらに考察する必要がある。必要に応じて、複数台の管理ホストまたはマルチプロセッサ搭載の Windows server 管理ホストを使って、各仮想マシンに割り当てる仮想プロセッサ数や主記憶容量を増やすなどの措置で対処していく方法も考えられる。この場合、付録 2 に示された自動化用スクリプト Copy-VM.ps1 に、より細かなパラメータの指定が必要になる。

仮想サーバと管理ホストとの間のデータ引き渡しのために、共通仮想ハードディスクを利用した。その方法では、仮想サーバと管理ホストとの両側での同時利用ができず、それぞれ利用前にマウントして、利用後にアンマウントするので非効率的である。現在の Linux 統合サービスが改良され Copy-VMFile が使えるまで待つか、別のより良い通信手段を考える必要がある。

今回は、仮想サーバの複製後の再設定作業では、ホスト名、IP アドレスと MAC アドレスの書き換えだけに限られた。電子メールサーバなどを含む演習の場合は、それぞれのサーバ設定ファイルの書き換え処理記述をス

クリプトに入れる必要がある。これについては、今後の課題の1つとして、いろいろなネットワーク演習課題の設計に合わせて、テンプレート作成および自動化スクリプト修正を行っていく予定である。

参考文献

(1) 山守一徳, 鷲尾 敦：教室端末での Hyper-V の活用方法　三重大学教育学部研究紀要 65 巻　印刷中.

(2) マイクロソフト社：Hyper-V, http://technet.microsoft.com/ja-jp/scriptcenter/powershell.aspx, 2013 年 11 月現在.

(3) マイクロソフト社：Windows PowerShell でのスクリプティング, http://technet.microsoft.com/ja-jp/scriptcenter/powershell.aspx, 2013 年 11 月現在.

(4) マイクロソフト社：Hyper-V Cmdlets in Windows PowerShell, http://technet.microsoft.com/jp-jp/library/hh848559.aspx, 2013 年 11 月現在.

付録1　/etc/rc.d/rc.localの最後に加える再設定処理の記述

```
if [ -e /etc/reconfig ]; then
  /bin/rm -f /etc/reconfig
  /bin/mount /dev/sdb1 /mnt/comm || exit
  dir=`/usr/bin/tr -d '¥r' < /mnt/comm/new-hostname` || exit
  /bin/rm -f /mnt/comm/new-hostname
  /usr/bin/tr -d '¥r' < /mnt/comm/$dir/ifcfg-eth0 >
              /etc/sysconfig/network-scripts/ifcfg-eth0
  /usr/bin/tr -d '¥r' < /mnt/comm/$dir/network >
             /etc/sysconfig/network
  /bin/rm -f /etc/udev/rules.d/70-persistent-net.rules
  /bin/umount /mnt/comm
  /sbin/poweroff
fi
```

付録2　PoweShell スクリプト Copy-VM.ps1 without param check

```
param(
    [Parameter(Mandatory=$True)]
    [string]$fromVM,
    [Parameter(Mandatory=$True)]
    [string]$startIPaddress,
    [string]$toPath = (Get-VMHost).VirtualHardDiskPath,
    [string]$commVHD = "$toPath¥comm.vhdx",
    [string]$prefix = "vm",
    [int]$suffix = 1,
    [int]$number = 1
 )
$ErrorActionPreference = "stop"
$hdd = (Get-VMHardDiskDrive "$fromVM").Path[0]
$switchname = (Get-VMNetworkAdapter "$fromVM").SwitchName
$frommacaddress=(Get-VMNetworkAdapter "$fromVM").MacAddress
$mac = $frommacaddress
foreach ($aa in (Get-VM | Get-VMNetworkAdapter).MacAddress){
       if( $aa -gt $mac ){ $mac = $aa } }
$domain=".edu.mie-u.ac.jp"
$ip = ([IPAddress]$startIPaddress).GetAddressBytes()

for ( $ii=0; $ii -Lt $number; ++$ii){
  $new = "$prefix" + ("{0:00}" -f ($suffix+$ii))
  $newpath = "$toPath¥$new"
  if( ! (Test-Path "$newpath" )) { mkdir "$newpath" }
  $newhdd = "$newpath¥$new.vhdx"
  $mac = "{0:x12}" -f ( 1 + "0x$mac" )
  $macSTR = $mac
  for($aa=10;$aa -gt 0;$aa-=2) {
      $macSTR=$macSTR.Insert($aa,":")}
  $ipSTR = $ip -join "." ; $ip[3]++;
  for($aa=3;$aa -gt 0;$aa--){
      if($ip[$aa] -ge 255){$ip[$aa]=0;$ip[$aa-1]++}
         else{break} }

  cp "$hdd" "$newhdd"
  New-VM "$new" -Path "$toPath" -VHDPath "$newhdd"
  Set-VMMemory "$new" -DynamicMemoryEnabled $true `
       -MinimumBytes 512MB -StartupBytes 512MB `
       -MaximumBytes 1GB
  Remove-VMNetworkAdapter -VMName "$new"
  Add-VMNetworkAdapter -VMName "$new" `
       -SwitchName "$switchname"
  Set-VMNetworkAdapter -VMName "$new" `
       -StaticMacAddress $frommacaddress
  Add-VMHardDiskDrive -VMName "$new" -Path "$commVHD" `
       -ControllerType SCSI -ControllerNumber 0 `
       -ControllerLocation 0
  $dl = (Mount-VHD -Path "$commVHD" -Passthru |
      get-disk | get-partition).DriveLetter + ":"
  write $new  | Out-File -FilePath "$dl¥new-hostname" `
      -Encoding ASCII
  if( ! (Test-Path "$dl¥$new" )) {mkdir "$dl¥$new" }
  cat "$dl¥$fromVM¥ifcfg-eth0" |
     %{$_ -replace "^HWADDR=.*", "HWADDR=$macSTR" } |
     %{$_ -replace "^IPADDR=.*","IPADDR=$ipSTR"} |
     Out-File -FilePath "$dl¥$new¥ifcfg-eth0" -Encoding ASCII
  cat "$dl¥$fromVM¥network" |
     %{$_ -replace "^HOSTNAME=.*", "HOSTNAME=$new$domain"} |
     Out-File -FilePath "$dl¥$new¥network"  -Encoding ASCII
  Dismount-VHD -Path "$commVHD"
  Start-VM "$new"
  while((Get-VM "$new").state -eq "Running"){sleep -s 5 }
  Remove-VMHardDiskDrive -VMName "$new" `
     -ControllerType SCSI -ControllerNumber 0 `
     -ControllerLocation 0
  Set-VMNetworkAdapter -VMName "$new" `
     -StaticMacAddress $macSTR
}
```