CLUSTERPRO® X

VMware vSphere システム構築ガイド

2016/6/1
第12版

CLUSTERPRO

改版履歴

版数	改版日付	内 容
1	2009/07/14	新規作成
2	2009/08/21	内部バージョン2.1.1-1に対応
3	2010/02/26	二つ目の仮想NICを追加する手順を記載 物理NICと仮想NICを結びつける手順を記載 vSphereのfirewall設定について記載 ホスト側NIC異常検出時の設定について記載 userwの設定について記載
4	2010/07/09	章立て等の文書構成を変更 リンク切れ、誤字脱字等の修正
5	2010/10/01	CLUSTERPRO X 3.0 に対応
6	2010/12/06	仮想ディスクによる共有ディスク型クラスタ構成の削除
7	2011/08/01	表記の誤りを修正
8	2011/10/11	CLUSTERPRO X 3.1 並びに VMware vSphere 5 に対応
9	2012/09/30	管理用OSの設定を追記
10	2012/12/10	VMware vSphere 5.1 に対応
11	2013/03/31	ゲストOS間クラスタの注意事項にシャットダウン時の注意事項を記載
12	2016/06/01	VMware vSphere 6.0 に対応

免責事項

本書の内容は、予告なしに変更されることがあります。
日本電気株式会社は、本書の技術的もしくは編集上の間違い、欠落について、一切責任をおいません。
また、お客様が期待される効果を得るために、本書に従った導入、使用および使用効果につきましては、
お客様の責任とさせていただきます。
本書に記載されている内容の著作権は、日本電気株式会社に帰属します。本書の内容の一部または全部
を日本電気株式会社の許諾なしに複製、改変、および翻訳することは禁止されています。

商標情報

CLUSTERPRO® X は日本電気株式会社の登録商標です。
AMD, AMD VirtualizationはAdvanced Micro Devices, Incの商標です。
Intel, Pentium, Xeon, Intel VTは、Intel Corporationの登録商標または商標です。
VMware, vMotionは、米国VMware, Inc.の登録商標です。
Linuxは米国及びその他の国におけるLinus Torvaldsの登録商標です。
その他のシステム名、社名、製品名等はそれぞれの会社の商標及び登録商標です。

目次

はじめに

対象読者と目的

『CLUSTERPROシステム構築ガイド』は、クラスタシステムに関して、システムを構築する管理者、およびユーザサポートを行うシステムエンジニア、保守員を対象にしています。
本書では、CLUSTERPRO環境下での動作確認が取れたソフトウェアをご紹介しています。ここでご紹介するソフトウェアや設定例は、あくまで参考情報としてご提供するものであり、各ソフトウェアの動作保証をするものではありません。

関連マニュアル

本書の利用にあたっては、必要に応じて以下のマニュアルを参照してください。

1. CLUSTERPROマニュアル
   CLUSTERPRO のマニュアルは、以下の 5 つに分類されます。

   『CLUSTERPRO X スタートアップガイド』(Getting Started Guide)
   CLUSTERPROを使用するユーザを対象読者とし、製品概要、動作環境、アップデート情報、既知の問題などについて記載します。

   『CLUSTERPRO X インストール & 設定ガイド』(Install and Configuration Guide)
   CLUSTERPRO を使用したクラスタ システムの導入を行うシステム エンジニアと、クラスタシステム導入後の保守・運用を行うシステム管理者を対象読者とし、CLUSTERPRO を使用したクラスタ システム導入から運用開始前までに必須の事項について説明します。実際にクラスタ システムを導入する際の順番に則して、CLUSTERPRO を使用したクラスタ システムの設計方法、CLUSTERPRO のインストールと設定手順、設定後の確認、運用開始前の評価方法について説明します。

   『CLUSTERPRO X リファレンス ガイド』(Reference Guide)
   管理者、およびCLUSTERPRO を使用したクラスタ システムの導入を行うシステム エンジニアを対象とし、CLUSTERPRO の運用手順、各モジュールの機能説明、メンテナンス関連情報およびトラブルシューティング情報等を記載します。『インストール & 設定ガイド』を補完する役割を持ちます。

   『CLUSTERPRO X 統合WebManager 管理者ガイド』(Integrated WebManager Administrator's Guide)
   CLUSTERPRO を使用したクラスタシステムを CLUSTERPRO 統合WebManager で管理するシステム管理者、および統合WebManager の導入を行うシステム エンジニアを対象読者とし、統合WebManager を使用したクラスタ システム導入時に必須の事項について、実際の手順に則して詳細を説明します。

   『CLUSTERPRO X WebManager Mobile 管理者ガイド』 (WebManager Mobile Administrator's Guide) - CLUSTERPRO X 3.1 以降
   CLUSTERPRO を使用したクラスタシステムを CLUSTERPRO WebManager Mobile で管理するシステム管理者、およびWebManager Mobile の導入を行うシステム エンジニアを対象読者とし、WebManager Mobile を使用したクラスタ システム導入時に必須の事項について、実際の手順に則して詳細を説明します。

   CLUSTERPRO マニュアルに関しては、以下を参照してください。
   『CLUSTERPRO Webサイト』
   http://jpn.nec.com/clusterpro/

2. VMware vSphereドキュメント
   VMware vSphereの詳細については、VMware vSphereドキュメントを参照してください。

   『VMware vSphereドキュメント』
   http://www.vmware.com/jp/support/pubs/vs_pubs.html

本書の構成

第 1 章　「構成」: VMware vSphere と CLUSTERPRO を組み合わせることにより実現可能なクラスタシステムについて記述します。

第 2 章　「動作環境」: VMware vSphere と CLUSTERPRO を組み合わせる場合の動作環境について記述します。

第 3 章　「注意事項」: VMware vSphere と CLUSTERPRO を組み合わせる場合の注意事項について記述します。

第 4 章　「仮想化環境構築手順」: VMware vSphere と CLUSTERPRO を組み合わせる場合の仮想化環境の構築手順について記述します。

第 5 章　「クラスタ環境構築手順」: VMware vSphere と CLUSTERPRO を組み合わせる場合のクラスタシステムの構築手順について記述します。

本書の表記規則

本書では、「注」および「重要」を以下のように表記します。

注: は、重要ではあるがデータ損失やシステムおよび機器の損傷には関連しない情報を表します。

重要: は、データ損失やシステムおよび機器の損傷を回避するために必要な情報を表します。

関連情報: は、参照先の情報の場所を表します。

また、本書では以下の表記法を使用します。

表記	使用方法	例
[ ] 角かっこ	コマンド名の前後 画面に表示される語 (ダイアログ ボックス、メニューなど) の前後	[スタート] をクリックします。 [プロパティ] ダイアログ ボックス
コマンドライン中の [ ] 角かっこ	かっこ内の値の指定が省略可能であることを示します。	`clpstat -s[-h host_name]`
モノスペースフォント (courier)	コマンド ライン、関数、パラメータ	`clpstat -s`
モノスペースフォント**太字** (courier)	ユーザが実際にコマンドプロンプトから入力する値を示します。	以下を入力します。 **`clpcl -s -a`**
モノスペースフォント (courier) 斜体	ユーザが有効な値に置き換えて入力する項目	`clpstat -s [-h host_name]`

本書で用いる用語

本書で用いる用語について説明します。

用語	略語	説明
物理サーバ	SV	VMware ESXまたは他のOSが動作しているサーバです。
単体OS	OS	仮想化基盤ではなく単独で利用する通常のOSです。
ホストOS	ホスト	仮想化基盤として物理サーバにインストールされているOS、つまりVMware ESXです。
仮想マシン	VM	ホストOS上に作成される仮想的なサーバまたはクライアントです。
ゲストOS	ゲスト	仮想マシンにインストールされているOSです。
管理用OS	←	VMware ESXiではService Consoleが提供されていません。管理用OSはService Consoleの代替としてホストOSを管理するためのゲストOSです。
CLUSTERPRO X	CLS	CLUSTERPRO Xです。
CLUSTERPRO X SingleServerSafe	SSS	CLUSTERPRO X SingleServerSafeです。
Application	AP	業務アプリケーションです。
VMware vSphere Management Assistant	vMA	ホストOSを管理するためにVMware社より提供されている管理用OSです。
VMware vSphere CLI (Command Line Interface)	CLI	ホストOSを管理、制御するために提供されているコマンドインターフェイスです。

第 1 章　構成

VMware vSphere と CLUSTERPRO を組み合わせることで、下記構成のクラスタを構築することができます。

ホストOS間クラスタ

VMware ESX の Service Console 上に CLUSTERPRO X をインストールし、物理サーバ同士でクラスタリングを行います。通常の業務アプリケーションのフェイルオーバのみならず、ゲスト OS をフェイルオーバさせることができます。

また、ゲストーホスト間の連携により、ゲスト OS 内の業務アプリケーションの監視も可能です。

クラスタ構築手順は『ホスト OS 間クラスタを構築する(34ページ)』を参照してください。また、ゲストーホスト連携を利用する場合は『ホスト OS 間クラスタでゲストーホスト連携を利用する(36ページ)』を参照してください。

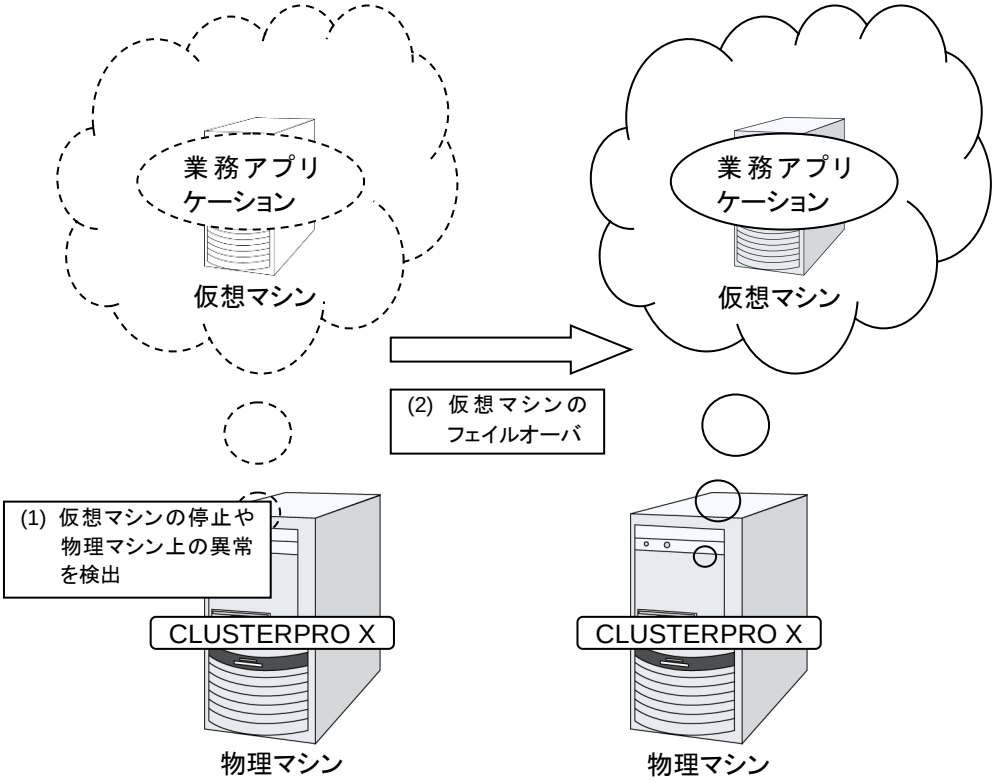


図 1 ： ホスト OS 間クラスタの概要図

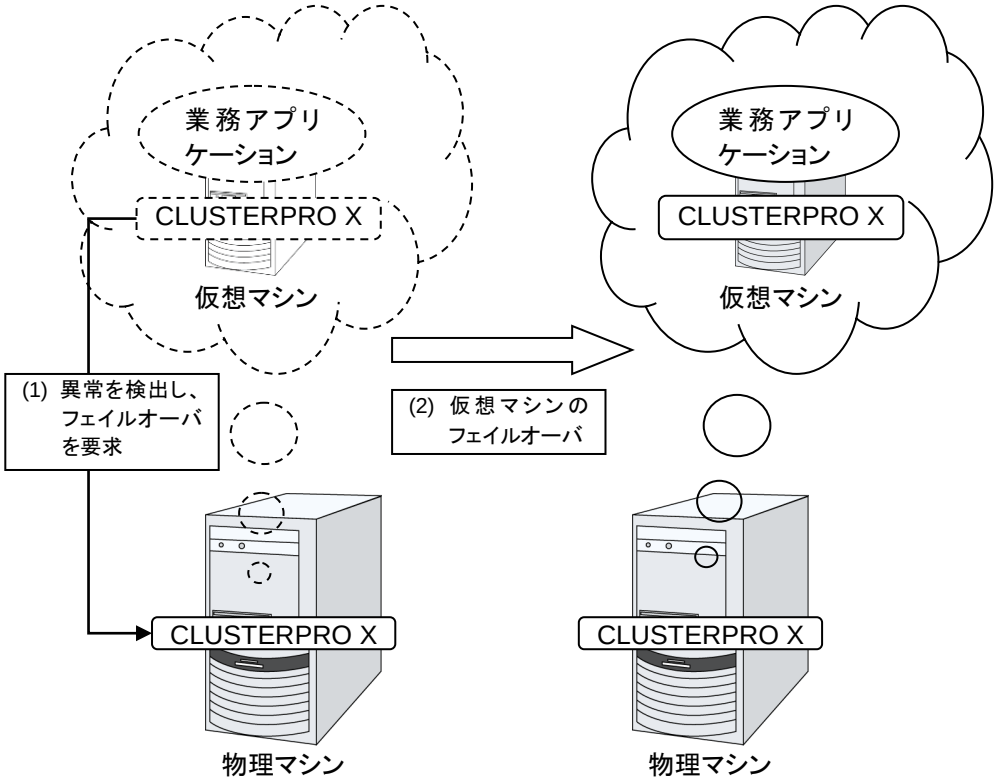


図 2 : ゲストーホスト連携時のホスト OS 間クラスタの概要図

ゲストOS間クラスタ

ゲスト OS 上に CLUSTERPRO X をインストールし、仮想マシン同士でクラスタリングを行います。通常のクラスタシステムと同様、業務アプリケーションのフェイルオーバが可能で、業務の可用性を高めることができます。

クラスタ構築手順は『ゲスト OS 間クラスタを構築する(37ページ)』を参照してください。

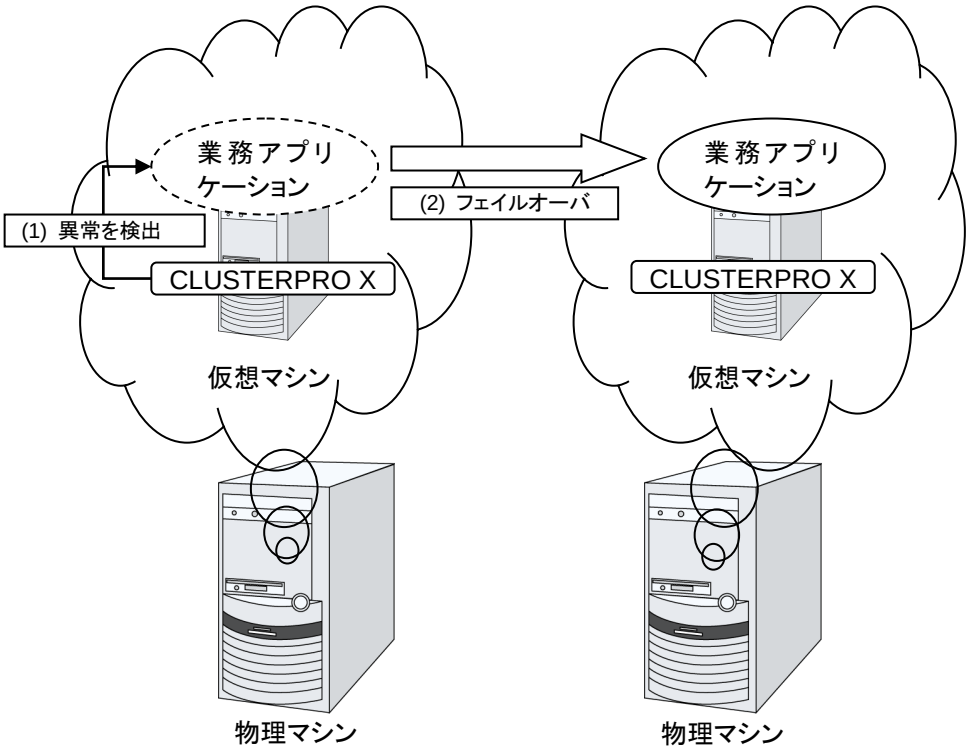


図 3 : ゲスト OS 間クラスタの概要図

管理用OS間クラスタ

VMware vSphere 5 で提供される HyperVisor は VMware ESXi に統合され、従来の VMware ESX で提供されていた Service Console は使用することができません。

Service Console の代替に、仮想マシンを管理するための管理用 OS(仮想マシン)を各ホストに用意します。この管理用 OS 上に CLUSTERPRO X をインストールすることで、VMware ESXi 環境でも通常の業務アプリケーションのみならず、ゲスト OS をフェイルオーバさせることができます。

また管理用 OS として vMA(VMware vSphere Management Assistant)を使用する場合、VMware ESXi を管理、制御するための Remote CLI 及び vSphere API を用いることで、Service Console と同等に物理マシンの監視も行うことが可能です。

クラスタ構築手順は『管理用 OS 間クラスタの構築(39ページ)』を参照してください。

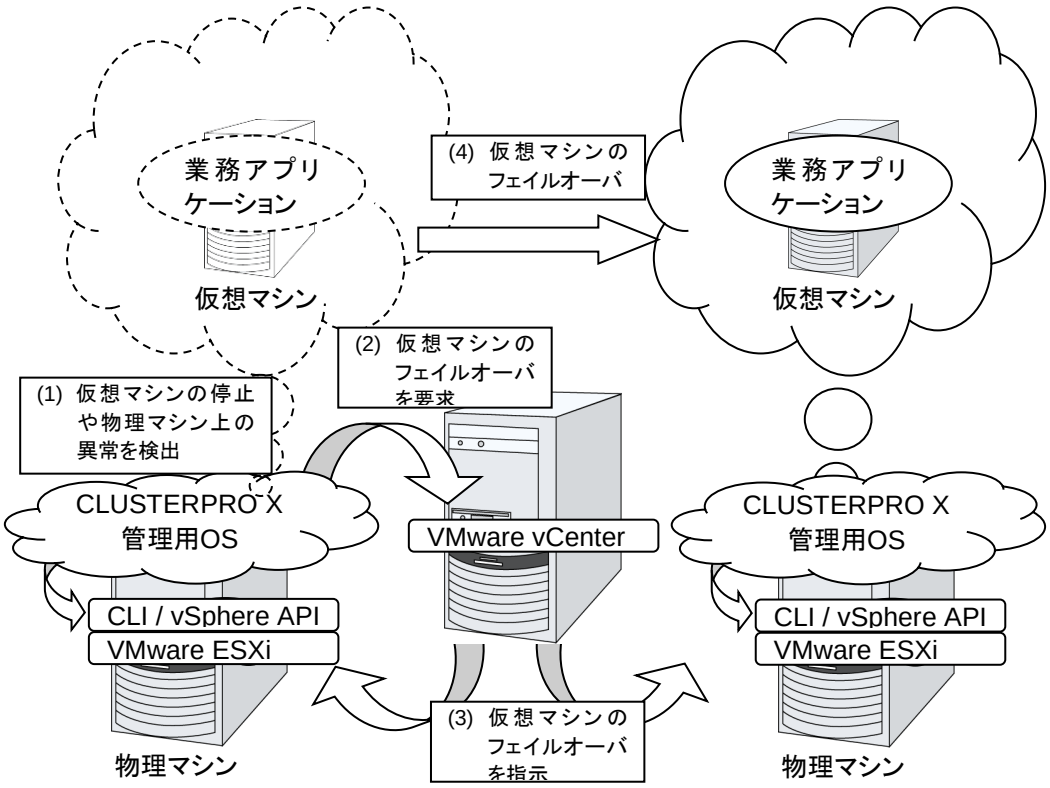


図 4 : 管理用 OS 間クラスタの概要図

物理サーバー仮想マシン間クラスタ

物理サーバ上で動作している OS と仮想マシン上で動作している OS に CLUSTERPRO X をインストールし、物理サーバと仮想マシン間でクラスタリングを行います。通常のクラスタシステムと同様、業務アプリケーションのフェイルオーバが可能です。

クラスタ構築手順は『物理サーバー仮想マシン間クラスタの構築(51ページ)』を参照してください。

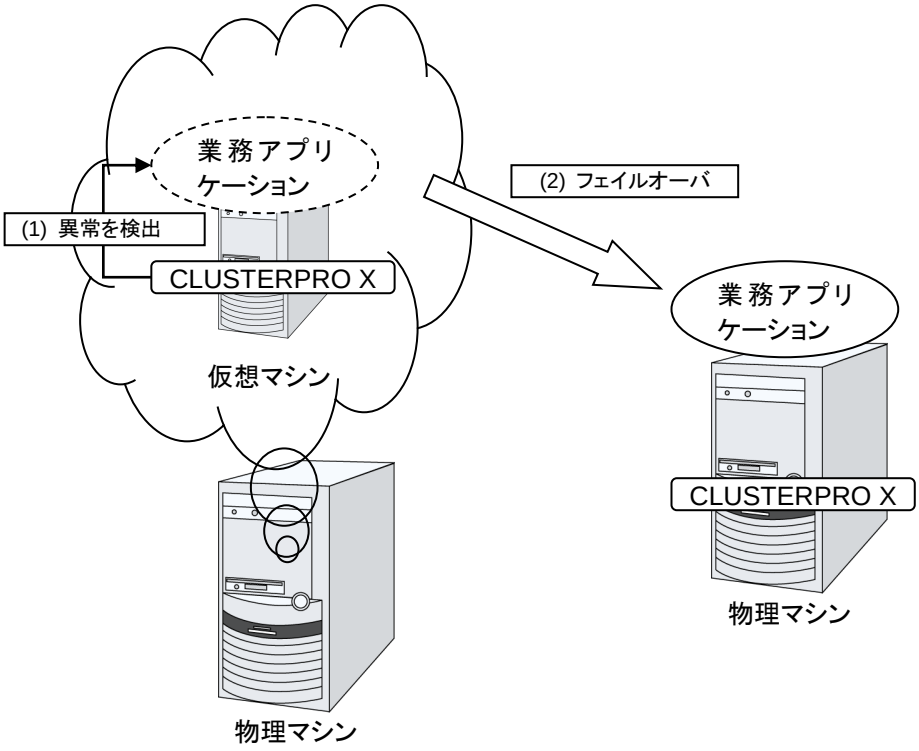


図 5 : 物理サーバー仮想マシン間クラスタの概要図

VMware HA連携

仮想マシンは VMware HA を構成し、ゲスト OS ダウン時／vCenter-ESX 間ネットワーク断絶時にはゲスト OS をフェイルオーバします。

VMware ESX の Service Console 上に CLUSTERPRO X SingleServerSafe をインストールすることで、VMware で検出できない物理 HW の異常を検出することができます。特に、ゲスト OS が格納される共有ディスクの異常検出時には速やかに ESX をシャットダウンし、動作が不安定なゲスト OS で業務を継続させないようにします。(図 6)

また、ゲスト OS 上に CLUSTERPRO X をインストールすることで、ゲスト OS 上の異常(仮想 HW 異常、AP 異常を含む)を検出し、業務をフェイルオーバすることが可能になります。(図 7)

本構成の構築手順は『VMware HA との連携(67ページ)』を参照してください。

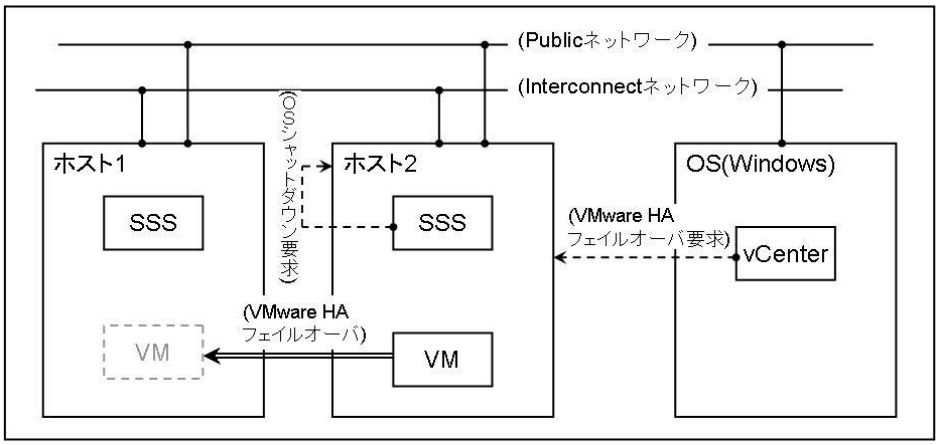


図 6 : ホスト上の CLUSTERPRO X SingleServerSafe と VMware HA の連携構成の概要図

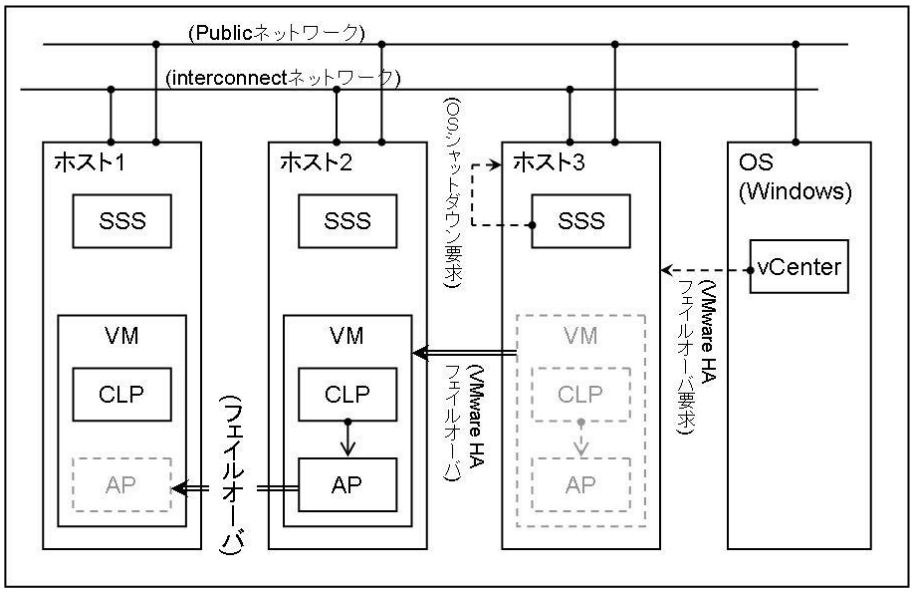


図 7 : ゲスト上の CLUSTERPRO X、ホスト上の CLUSTERPRO X SingleServerSafe と VMware HA の連携構成の概要図

第 2 章　動作環境

本書が対象とするVMware vSphere及びCLUSTERPROのバージョンは下記のとおりです。

表 1 構成一覧

VMware vSphere バージョン	クラスタ構成	動作確認済み CLUSTERPRO バージョン	構築手順
VMware vSphere 4.0 VMware vSphere 4.0 update 1 VMware vSphere 4.0 update 2 VMware vSphere 4.1 VMware vSphere 4.1 update 1 VMware vSphere 4.1 update 2	ホストOS間クラスタ	CLUSTERPRO X 2.1 for Linux	ホストOS間クラスタの構築(30ページ) 仮想マシンの制御、監視にはスクリプトを使用する必要があります。
		CLUSTERPRO X 3.0 for Linux CLUSTERPRO X 3.1 for Linux	ホストOS間クラスタの構築(30ページ)
	ゲストOS間クラスタ	CLUSTERPRO X 2.1 for Windows/Linux CLUSTERPRO X 3.0 for Windows/Linux CLUSTERPRO X 3.1 for Windows/Linux	ゲストOS間クラスタの構築(37ページ)
VMware vSphere 5 VMware vSphere 5.1 VMware vSphere 5.5	ゲストOS間クラスタ	CLUSTERPRO X 2.1 for Windows/Linux CLUSTERPRO X 3.0 for Windows/Linux CLUSTERPRO X 3.1 for Windows/Linux CLUSTERPRO X 3.2 for Windows/Linux CLUSTERPRO X 3.3 for Windows/Linux	ゲストOS間クラスタの構築(37ページ)
	管理用OS間クラスタ	CLUSTERPRO X 3.1 for Linux CLUSTERPRO X 3.2 for Linux CLUSTERPRO X 3.3 for Linux ＊：VMware vSphere 5.5 は CLUSTERPRO X 3.2以降のみ対象	管理用OS間クラスタの構築(39ページ) ホストのハードウェア監視をするには管理用OSにVMware vSphere Management Assistant を使用し、本書のサンプルスクリプトを使用する必要があります。
VMware vSphere 6.0	ゲストOS間クラスタ	CLUSTERPRO X3.3 for Windows/Linux	ゲストOS間クラスタの構築(37ページ)

第 3 章　注意事項

Service ConsoleでCLUSTERPROを利用する場合の注意事項

- Service Console には、x86_64 版の rpm (clusterpro-X.X.X-X.x86_64.rpm) をインストールしてください。[1]
- ユーザ空間監視リソース (userw) やシャットダウンストール監視を利用する場合、監視方式は keepalive を選択してください。
- 下記の機能を利用することはできません。
  - ミラーディスクリソース(md)
  - ハイブリットディスクリソース(hd)
- Service Console で CLUSTERPRO を利用する場合、CLUSTERPRO の通信用ポートにアクセスできるようにファイアウォールを設定する必要があります。
- 二つ目の NIC を使用する場合、ホストのネットワーク構成で「サービス コンソール ポート」の追加が必要です。
- ホストのネットワーク構成の各ポートに共通の IP アドレスを設定することはできません。「VMkernel」と「Service Console」を同時に使用したい場合は別々の IP アドレスを設定してください。
- VMware ESX の core の出力先は、デフォルトでは/var/core 配下となるため、CLUSTERPRO のログ収集で収集できません。
- VMware vSphere 5 では利用できる HyperVisor が、VMware ESXi 5.0 のみとなるため、Service Console の利用はできません。

仮想マシンでCLUSTERPROを利用する場合の注意事項

- 二つ目の物理アダプタを利用したい場合、ホストのネットワーク構成で二つ目の物理アダプタに結びついた仮想スイッチを作成する必要があります。
- NIC Link Up/Down モニタリソースは使用できません。代用として IP モニタリソースを使用してください。

[1] X2.1にはVMware ESX 3.5 Service Console用のrpm(clusterpro-2.1.X-X.vmware.i386.rpm)があるため、間違って利用しないよう注意してください。

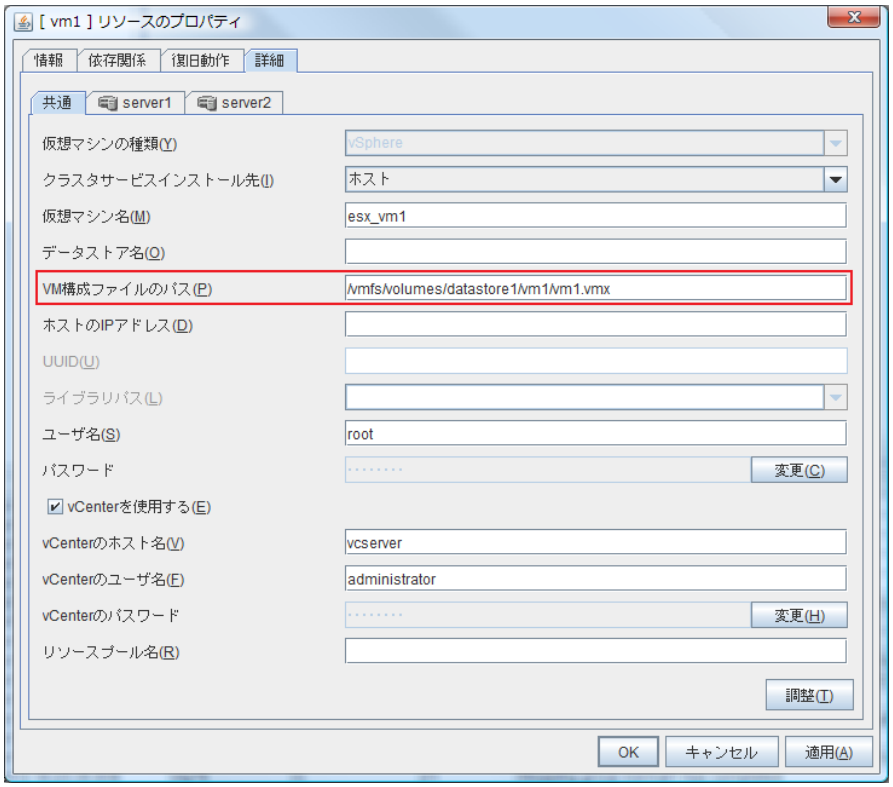
管理用OSでCLUSTERPROを利用する場合の注意事項

- CLUSTERPRO X 3.1 以降でのみ利用可能です。
  - vMA には、x86_64 版の rpm をインストールしてください。
- 管理用 OS に Red Hat Enterprise Linux 6 の使用可否
  - 内部バージョン 3.1.4 以前：使用不可
  - 内部バージョン 3.1.5 以降：使用可
- 管理用 OS として検証済みの OS は以下の通りです。
  - VMware vSphere Management Assistant 5.0
  - VMware vSphere Management Assistant 5.1
  - Red Hat Enterprise Linux 5.5 （IA32 版 / x86_64 版 ）

  内部バージョン 3.1.5 以降：
  - Red Hat Enterprise Linux 6.0 （IA32 版 / x86_64 版 ）
- ホスト監視用モニタの設定(45ページ)を行う場合、使用する管理用 OS は vMA である必要があります。
- 二つ目の物理アダプタを利用したい場合、ホストのネットワーク構成で二つ目の物理アダプタに結びついた仮想スイッチを作成する必要があります。
- 仮想マシンリソースで制御する仮想マシンは仮想マシン名とデータストア上のディレクトリ名が同一である必要があります。仮想マシン名の変更(27ページ)を参照してください。

ホストOS間クラスタの注意事項

- クラスタ運用時に CLUSTERPRO の管理対象になっている仮想マシンの[サスペンド]を行わないでください。
- ゲストーホスト連携を利用する場合は、ゲスト OS に CLUSTERPRO X2.1 以降の製品をインストールする必要があります。
- 仮想マシンリソースの設定で必ず『VM 構成ファイルのパス』を設定してください。



ゲストOS間クラスタの注意事項

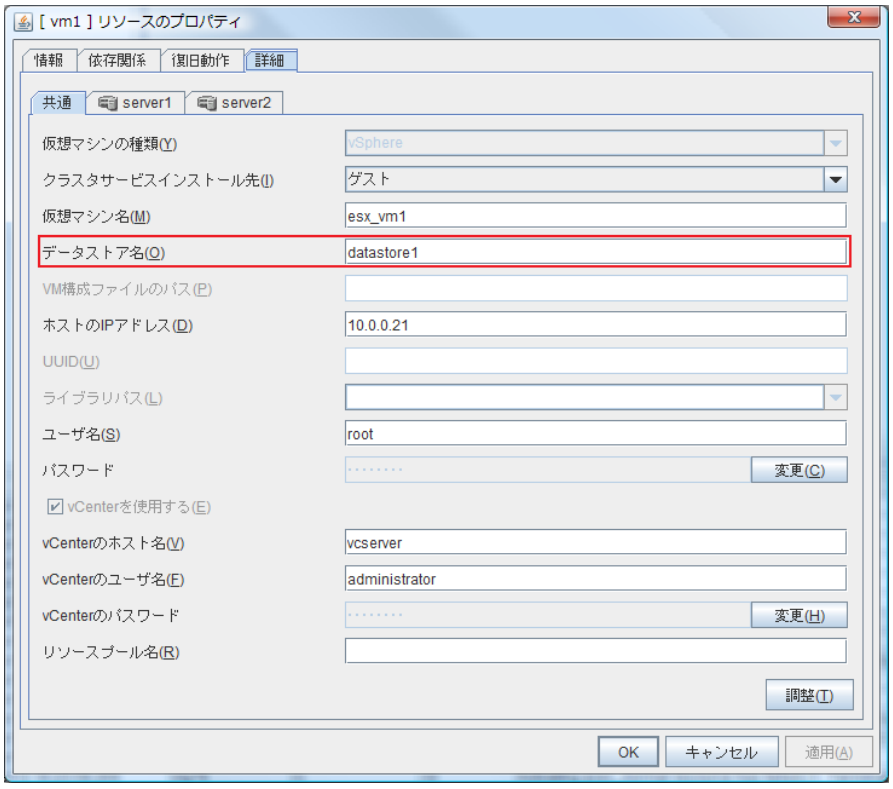
- COM ハートビートは使用できません。
- IPMI の機能を使用する下記の機能を使用することができません。
  - 強制停止機能
  - 筐体 ID ランプ連携機能
  - ユーザ空間監視の監視方法 ”ipmi”による監視
  - シャットダウンストール監視の監視方法”ipmi”による監視
  - グループリソースの活性異常/非活性異常時の最終動作
    “BMC Reset”, “BMC Power Off”, “BMC Power Cycle”, “BMC NMI”
  - モニタリソースの異常検出時の最終動作
    “BMC Reset”, “BMC Power Off”, “BMC Power Cycle”, “BMC NMI”
- 共有ディスクを利用する場合、下記の条件を満たす必要があります。
  - 共有ディスクが「Raw デバイスのマッピング」であること。
  - 「Raw デバイスマッピング」に指定するデバイスは iSCSI 接続のディスクは指定しないでください。
  - 仮想マシンの SCSI コントローラの「SCSI バスの共有」設定が「物理」または「仮想」であること。「仮想」を選ぶと 図 3 のような ESX ホストを跨いだゲスト OS 間クラスタは構成できません。
    「物理」: あらゆる ESX ホストの VM 間でハードディスクが共有できる
    「仮想」: 同一 ESX ホストの VM 間でハードディスクが共有できる
- クラスタ運用時には、仮想マシンの[サスペンド]を行わないでください。仮想マシンの[サスペンド]を行うと、CLUSTERPRO がハートビートタイムアウトを検出し、他のサーバでフェイルオーバグループを起動します。この状態で、[サスペンド]していた仮想マシンを[リジューム]すると、両系活性状態となり、データ保護の観点からそのフェイルオーバグループが起動している両方の仮想マシンをシャットダウンします。
- UPS 連携などで CLUSTERPRO 管理外からゲスト OS のシャットダウンが実行される場合、クラスタが正常に停止されないため、次回起動時にミラー再同期等が必要になってしまう場合があります。
  シャットダウン時にクラスタを停止するようパワーオフスクリプトの設定(26ページ)を参照し、設定を行ってください。

管理用OS間クラスタの注意事項

- 管理用 OS は管理対象の VMware ESXi と一対一の構成でインストールする必要があります。
- 仮想マシンリソースを利用する場合、VMware vCenter を用意する必要があります。
- COM ハートビートは使用できません。
- IPMI の機能を使用する下記の機能を使用することができません。
  - ユーザ空間監視の監視方法 ”ipmi”による監視
  - シャットダウンストール監視の監視方法”ipmi”による監視

- グループリソースの活性異常/非活性異常時の最終動作
  “BMC Reset”, “BMC Power Off”, “BMC Power Cycle”, “BMC NMI”
- モニタリソースの異常検出時の最終動作
  “BMC Reset”, “BMC Power Off”, “BMC Power Cycle”, “BMC NMI”

- クラスタ運用時には、管理用 OS がインストールされた仮想マシンの[サスペンド]を行わないでください。仮想マシンの[サスペンド]を行うと、CLUSTERPRO がハートビートタイムアウトを検出し、他のサーバでフェイルオーバグループを起動します。この状態で、[サスペンド]していた仮想マシンを[リジューム]すると、両系活性状態となり、データ保護の観点からそのフェイルオーバグループが起動している両方の仮想マシンをシャットダウンします。
- モニタリソースの回復動作などで OS シャットダウンを選択している場合、シャットダウンされるのはホストではなく、管理用 OS になります。
- 管理用 OS に vMA を使用している場合、本書のサンプルスクリプトshutdownhost.pl (90 ページ)を使用することでホストのシャットダウンを実行することが可能です。
- シャットダウンやパニック、パワーオフなどにより、管理用 OS がダウンした時に、ホストが健在な場合は、管理用 OS が動作していたホスト上で動作しているその他のゲストはライブマイグレーションが実行され、待機系サーバに移行します。
- クラスタ運用時に CLUSTERPRO の管理対象になっている仮想マシンの[サスペンド]を行わないでください。
- 仮想マシンリソースの設定で必ず『データストア名』を設定してください。



物理サーバー仮想マシン間クラスタの注意事項

- 「ゲスト OS 間クラスタの注意事項」を参照してください。

vMotionを併用する場合の注意事項

- vMotion を利用する場合、下記の条件を満たす必要があります。
  - VMware vCenter Server が導入されていること。
  - 各 ESX のネットワークに vMotion で使用できる「VMkernel ポート」が構成されていること。
  - 共有ストレージ装置[1]が構成されており、仮想マシンが共有ストレージ上のデータストアに格納されていること。
  - VMware vSphere 5.5 以前においては、仮想マシンの SCSI コントローラの「SCSI バスの共有」設定が「なし」であること。
    VMware vSphere 6.0 においては、仮想マシンの SCSI コントローラの「SCSI バスの共有」設定が「なし」であり、また、仮想マシン間で共有する共有ディスクについては、SCSI コントローラの「SCSI バスの共有」設定が「物理」であること。
  - DNS が正しく構成されており、vCenter と全 ESX のサービスコンソールは互いに名前解決ができること。
  - 各 ESX に構成されているホスト名と、DNS 側にて登録されているホスト名が一致していること。
- VMware vSphere 5.5 以前においては、vMotion は、共有ディスクを利用するゲスト OS 間クラスタとは併用できません。VMware vSphere の仕様により、仮想マシンの設定で SCSI コントローラの「SCSI バスの共有」を「なし」にすると、同じハードディスクを持つ仮想マシンは一つしか起動できなくなります。

表 2 : vMotion 併用可否一覧

クラスタ構成	**ホスト間クラスタ、管理用OS間クラスタ、**VMwareHAなど			**ゲストOS間クラスタ**	
vMotion対象	非クラスタ構成のゲストOS	**ゲストOS間クラスタ**構成のゲストOS		クラスタを構成するゲストOS	
		共有ディスク型	ミラーディスク型	共有ディスク型	ミラーディスク型
vMotion併用可否	可	不可	可	VMware vSphere 5.5 以前においては不可 VMware vSphere 6.0 においては可	可

※ **太字**はCLUSTERPROによるクラスタ構成

[1]共有ストレージ装置との接続方式として Fibre Channel SAN、iSCSI SANもしくはNFSが利用可能です。

VMware HAを併用する場合の注意事項

- VMware HA を利用する場合、下記の条件を満たす必要があります。
  - VMware vCenter Server が導入されていること。
  - VMware HA を利用できるライセンスを保有していること。
  - DNS が正しく構成されており、vCenter と全 ESX のサービスコンソールは互いに名前解決ができること。
  - 各 ESX に構成されているホスト名と、DNS 側にて登録されているホスト名が一致していること。
  - 各 ESX において、Service Console の Gateway アドレスが正しく設定されており、Gateway が ping に応答すること。
  - VMware HA で仮想マシンの監視を有効にする場合、監視する仮想マシンに VMware Tools がインストールされていること。
  - 管理用 OS は対象外に設定すること。

第 4 章　仮想化環境構築手順

ホストOS間クラスタ用の設定

VMware ESX のインストール

VMware 社が用意しているインストールガイドにしたがって VMware ESX をインストールしてください。

- VMware サポートのリソース ドキュメント
  http://www.vmware.com/jp/support/pubs/

ファイアウォールの設定

Service Console でCLUSTERPROを利用する場合は、CLUSTERPROの通信用ポートにアクセスできるようにファイアウォールを設定してください。

ファイアウォールを無効化する場合:

```
# esxcfg-firewall --allow Incoming --allow Outgoing
```

CLUSTERPRO が利用するポート番号は下記を参照してください。

- CLUSTERPRO X for Linux スタートアップガイド
  第 5 章 注意制限事項
  > OS インストール後、CLUSTERPRO インストール前
  >> 通信ポート番号

仮想マシンの設定

仮想マシンの設定(28ページ)を参照し、仮想マシンリソースで制御する仮想マシンの設定をしてください。

仮想スイッチの設定

仮想スイッチの追加（28ページ）を参考に必要に応じて仮想スイッチを設定してください。

サービスコンソールポートを追加

二つ目の仮想スイッチを Service Console で利用する場合は、「サービスコンソールポート」を追加してください。
注:「VMkernel 用」と「Service Console 用」の IP は同じ IP に設定できません。

(1) 二つ目の仮想スイッチの「プロパティ」をクリックし、「ポート」タブを選択します。

(2) 「追加」をクリックし、「サービス コンソール」を選択し、「次へ」をクリックします。

(3) ポートグループのプロパティの「ネットワーク ラベル」で任意の名前を入力し、「次へ」をクリックします。

(4) 「次の IP 設定を使用」を選択し、「IP アドレス」と「サブネットマスク」を記入し、「次へ」をクリックします。

(5) 設定を確認し、「終了」をクリックします。

(6) 仮想スイッチの「プロパティ」の「閉じる」をクリックします。

ゲストOS間クラスタ用の設定

仮想スイッチの設定

仮想スイッチの追加（28ページ）を参考に必要に応じて仮想スイッチを設定してください。

仮想マシンの作成

構築するクラスタシステムにあわせて仮想マシンを作成してください。下記に仮想マシンの作成例を記載しますので参考にしてください。

仮想マシンの作成手順の詳細については、VMware 社が用意している基本システム管理ガイドを参照してください。

- VMware サポートのリソース ドキュメント
  http://www.vmware.com/jp/support/pubs/

共有ディスクを利用する仮想マシンを作成する[1]

(1) 「新規仮想マシンの作成」を実行します。

(2) 「構成」で「標準」を選択し、「次へ」をクリックします。

(3) 任意の名前を入力し、「次へ」をクリックします。

(4) 適当なホストを選択し、「次へ」をクリックします。

(5) 適当なデータストアを選択し、「次へ」をクリックします。

(6) インストールするゲスト OS を選択し、「次へ」をクリックします。

(7) 任意の仮想ディスクサイズを指定し、「次へ」をクリックします。ここでは OS をインストールするディスクのサイズを指定します。クラスタシステムの共有ディスク用の設定は後ほど行います。

(8) 設定を確認し、「終了」をクリックします。

(9) 追加した仮想マシンを右クリックし、「設定の編集」を実行する。

(10) 「ハードウェア」タブの「追加」をクリックします。

(11) 二つ目の VM Network を追加する必要がある場合、「イーサネット アダプタ」を選択し、「次へ」をクリックします。

(12) 「アダプタ タイプ」を設定し、「ネットワーク ラベル」で「二つ目の仮想スイッチのネットワーク ラベル」を選択し、「次へ」をクリックします。

(13) 設定を確認し、「終了」をクリックします。

(14) 「ハードディスク」を選択し、「次へ」をクリックします。

(15) 「Raw デバイスのマッピング」を選択し、「次へ」をクリックします。
なお、2 台目以降の仮想マシンの場合は、「既存の仮想ディスクを使用」を選択し、「次へ」をクリック、1 台目で作成した「*.vmdk」ファイルを選択してください。

(16) 利用するディスク LUN を指定し、「次へ」をクリックします。

[1] 仮想マシンに共有ディスクを設定すると、vMotionが利用できなくなる場合があります。詳しくは『vMotion を併用する場合の注意事項(22ページ)』を参照してください。

(17) 共有ディスクの「SCSI コントローラ」を分けるため、「仮想デバイスノード」の設定で、「SCSI (X:Y)」または「IDE (X:Y)」の X の値が OS インストール用のディスクとは異なるものを選択し、「次へ」をクリックします。

(18) 設定を確認し、「終了」をクリックします。

(19) ハードウェアのリストに「ハードディスク」以外に「SCSI コントローラ」も追加されていることを確認してください。もし「SCSI コントローラ」が追加されていない場合は、追加した「ハードディスク」の「仮想デバイスノード」の設定が間違っている可能性があります。

(20) vMotionを併用しない場合は、「SCSIコントローラ」を選択すると表示される「SCSIバスの共有」を「物理」または「仮想」に変更し、「OK」をクリックしてください。vMotion を併用する場合は、「物理」に変更し、「OK」をクリックしてください。

(21) 以上で共有ディスクを利用する仮想マシンが作成されました。OS をインストールしていなければインストールしてください。

ミラーディスクを利用する仮想マシンを作成する

(1) 『共有ディスクを利用する仮想マシンを作成する』の(14)まで同様に作業します。

(2) 「新規仮想ディスクを作成」を選択し、「次へ」をクリックします。2 台目以降も「新規仮想ディスクを作成」を選択してください。

(3) 「新規仮想ディスクを作成」を選択した場合は仮想ディスクの容量を指定し、「次へ」をクリックします。全ての仮想マシンで同じ容量にすることを推奨します。

(4) そのまま、「次へ」をクリックします。

(5) 設定を確認し、「終了」をクリックします。

(6) 以上でミラーディスクを利用する仮想マシンが作成されました。OS をインストールしていなければインストールしてください。

パワーオフスクリプトの設定

CLUSTERPRO管理外からゲストOSがシャットダウンされるとクラスタが正常に停止されないため、次回起動時にミラー再同期等が必要になる場合があります。
これを防止するために、VMware Toolsのカスタムスクリプトで、シャットダウン前にクラスタの停止コマンドを実行するよう設定を行うことを推奨します。
カスタムスクリプトの詳細・設定方法についてはVMware社が提供しているドキュメント『VMware Toolsのインストールと構成』
http://www.vmware.com/jp/support/support-resources/pubs/vsphere-esxi-vcenter-server-pubs/
の『VMware Tools構成ユーティリティの使用 – VMware Toolsのカスタムスクリプトの使用』を参照してください。

(1) パワーオフ操作時に実行されるスクリプトを編集します。

(2) クラスタ停止コマンドを実行するようスクリプトに以下を追記します。

```
clpcl -t
```

管理用OS間クラスタ用の設定

VMware ESXi のインストール

VMware 社が用意しているインストールガイドにしたがって VMware ESXi をインストールしてください。

- VMware サポートのリソース ドキュメント
  http://www.vmware.com/jp/support/pubs/

管理用OS のインストール

管理用 OS に vMA を使用する場合は、VMware 社が用意しているインストールガイドにしたがって vMA をインストールしてください。

- VMware サポートのリソース ドキュメント
  http://www.vmware.com/jp/support/pubs/

その他の OS を使用する場合は、CLUSTERPRO がサポートするゲスト OS をインストールしてください。

SSH認証の設定

管理用 OS から仮想マシンを制御できるように設定するため、管理用 OS とホストが鍵認証方式で SSH 接続できるように設定をしてください。

(1) ホストの SSH 認証を有効に設定します。

(2) 各管理用 OS で SSH 認証の鍵を生成します。

(3) 各管理用 OS が動作するホストに上記で作成した鍵を追加します。

(4) 各管理用 OS からホストに鍵認証方式で SSH 接続できることを確認してください。

仮想スイッチの設定

仮想スイッチの追加（28ページ）を参考に必要に応じて仮想スイッチを設定してください。

仮想マシンの設定

仮想マシンの設定(28ページ)を参照し、仮想マシンリソースで制御する仮想マシンの設定をしてください。

仮想マシン名の変更

仮想マシンリソースで制御する仮想マシンの仮想マシン名を変更する場合、以下の手順で変更を行ってください。

(1) VMware vSphere Client から接続し、仮想マシンを選択し、右クリックメニューから「名前の変更」をクリックします。

(2) 仮想マシンの名前を変更します。

(3) 「インベントリ」 - 「データストアおよびデータストア クラスター」を開きます。

(4) 変更したい仮想マシンが存在するデータストアを選択します。

(5) 「このデータストアを参照する」をクリックし、「データストアブラウザ」を開きます。

(6) 変更したい仮想マシンの構成ファイルが存在するディレクトリを選択し、右クリックメニューから「名前の変更」をクリックします。

(7) (2)で変更した名前と同一の名前に変更します。

(8) 「Cluster Manager」の「設定モード」より「仮想マシンリソース」の「仮想マシン名」を変更した仮想マシン名に変更します。

仮想マシンの設定

仮想マシンリソースで制御する仮想マシンに、以下の手順に従って、構成パラメータを追加してください。

(1) vSphere Client から対象の仮想マシンの「仮想マシンのプロパティ」画面を開きます。

(2) 「オプション」タブの左ペインより「詳細」 - 「全般」を選択し、右下の「構成パラメータ」をクリックします。

「構成パラメータ」画面の「行の追加」をクリックし、以下のパラメータを入力してください。
- 名前 : answer.msg.uuid.altered
- 値 : I moved it

仮想スイッチの追加

サーバに複数の物理 NIC がある場合、デフォルトでは一方に関連付けられた仮想スイッチのみ作成されるため、もう一方の物理 NIC を利用する場合は新たに仮想スイッチを追加する必要があります。

二つ目の仮想スイッチを追加する

(1) VMware vSphere Client から接続し、ホスト（物理サーバ）を選択し、「構成」タブの「ネットワーク」をクリックします。

(2) 「ネットワークの追加」をクリックします。

(3) 「仮想マシン」を選択し、「次へ」をクリックします。

(4) 「仮想スイッチの作成」を選択し、「次へ」をクリックします。

(5) ポットグループのプロパティの「ネットワーク ラベル」で任意の名前を入力し、「次へ」をクリックします。

(6) 設定を確認し、「終了」をクリックします。

VMkernelポートを追加する

二つ目の仮想スイッチを vMotion で利用する場合は、「VMkernel ポート」を追加してください。

(1) 二つ目の仮想スイッチの「プロパティ」をクリックし、「ポート」タブを選択します。

(2) 「追加」をクリックし、「VMkernel」を選択し、「次へ」をクリックします。

(3) ポットグループのプロパティの「ネットワーク ラベル」で任意の名前を入力し、「このポートグループを vMotion で使用」をチェック入れて、「次へ」をクリックします。

(4) 「次の IP 設定を使用」を選択し、「IP アドレス」と「サブネットマスク」を記入し、「次へ」をクリックします。

(5) 設定を確認し、「終了」をクリックします。

(6) 「デフォルトゲートウェイが設定されていません。このネットワークインターフェースを使用するには、デフォルトゲートウェイの設定が必要になることがあります。今すぐ構成しますか?」という警告が出力された場合、「いいえ」をクリックします。

(7) 仮想スイッチの「プロパティ」の「閉じる」をクリックします。

第 5 章　　クラスタ環境構築手順

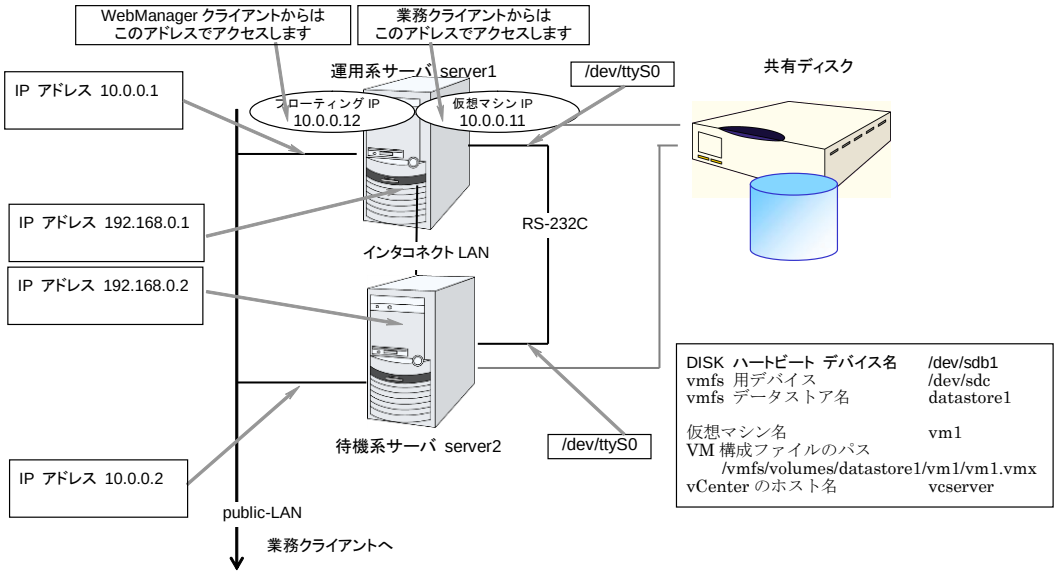
ホストOS間クラスタの構築

ホストOS間クラスタ設定例

ホスト OS 間クラスタを構築する場合、ホスト OS の CLUSTERPRO のバージョンによって使用可能な仮想マシンの制御方法が異なります。

表 3 ： ホスト OS 間クラスタの構築対応表

	仮想マシンの制御方法	CLUSTERPRO X2.1 for Linux	CLUSTERPRO X3.x for Linux
パターン 1	EXEC リソースとカスタムモニタリソースを使用する	○	○
パターン 2	仮想マシンリソースと仮想マシンモニタリソースを使用する	×	○



2 ノードの構成設定例

設定対象	設定パラメータ	設定値 (パターン1)	設定値 (パターン2)
クラスタ構成	クラスタ名	cluster	←
	サーバ数	2	←
	フェイルオーバ グループ数	2	←
	モニタ リソース数	5	←
ハートビート リソース	LAN ハートビート数	2	←
	COM ハートビート数	1	←
	ディスク ハートビート数	1	←

設定対象	設定パラメータ	設定値 (パターン1)	設定値 (パターン2)
1 台目のサーバの情報 (マスタ サーバ)	サーバ名*1	server1	←
	インタコネクトの IP アドレス (専用)	192.168.0.1	←
	インタコネクトの IP アドレス (バックアップ)	10.0.0.1	←
	パブリックの IP アドレス	10.0.0.1	←
	COM ハートビート デバイス	/dev/ttyS0	←
	ディスク ハートビートデバイス	/dev/sdb1	←
2 台目のサーバの情報	サーバ名*1	server2	←
	インタコネクトの IP アドレス (専用)	192.168.0.2	←
	インタコネクトの IP アドレス (バックアップ)	10.0.0.2	←
	パブリックの IP アドレス	10.0.0.2	←
	COM ハートビート デバイス	/dev/ttyS0	←
	ディスク ハートビートデバイス	/dev/sdb1	←
管理用のグループ	タイプ	フェイルオーバ	←
(WebManager 用)	グループ名	ManagementGroup	←
	起動サーバ	全てのサーバ	←
	グループ リソース数	1	←
管理用グループのグループリソース *2	タイプ	floating ip resource	←
	グループ リソース名	ManagementIP	←
	IPアドレス	10.0.0.12	←
業務用のグループ	タイプ	フェイルオーバ	仮想マシン
	グループ名	failover-vm	virtualmachine1
	起動サーバ	全てのサーバ	←
	グループ リソース数	1	←
1 つ目のグループリソース （仮想マシン制御用リソース）*3	タイプ	execute resource	virtual machine resource
	グループ リソース名	exec-vm	vm1
	スクリプト	標準スクリプト	ー
	仮想マシンの種類	ー	vSphere
	クラスタサービスインストール先	ー	ホスト
	仮想マシン名	ー	esx_vm1
	VM構成ファイルのパス	ー	/vmfs/volumes/datastore1/vm1/vm1.vmx
	vCenterを使用する	ー	オン

設定対象	設定パラメータ	設定値 (パターン1)	設定値 (パターン2)
	vCenterのホスト名	ー	vcserver
	リクエストタイムアウト	ー	120
	仮想マシン起動待ち時間	ー	30
1 つ目のモニタリソース	タイプ	user mode monitor	←
(デフォルト作成)	モニタ リソース名	userw	←
2 つ目のモニタリソース	タイプ	diskw monitor	←
	モニタ リソース名	diskw1	←
	監視対象	/dev/sdc	←
	監視方法	TUR	←
	異常検出時	クラスタ デーモン停止と OS シャットダウン	←
3 つ目のモニタリソース	タイプ	NIC Link Up/Down monitor	←
	モニタ リソース名	miiw1	←
	監視対象	vmnic0 (Public LANの仮想スイッチインタフェース)	←
	異常検出時	“ManagementGroup” グループのフェイルオーバ	←
4 つ目のモニタリソース	タイプ	NIC Link Up/Down monitor	←
	モニタ リソース名	miiw2	←
	監視対象	vmnic0 (Public LANの仮想スイッチインタフェース)	←
	回復動作	ー	回復対象に対してフェイルオーバ実行
	フェイルオーバ実行前にマイグレーションを実行する	ー	オン
	異常検出時	“failover-vm” グループのフェイルオーバ	←
5 つ目のモニタリソース	タイプ	Custom monitor	virtual machine monitor
（仮想マシン監視用モニタ）*3*4	モニタ リソース名	genw-vm	vmw1
	スクリプト	標準スクリプト	ー
	仮想マシンリソース	ー	vm1
	外部マイグレーション発生時の待ち時間	ー	15(秒)
	監視タイミング	活性時	常時

設定対象	設定パラメータ	設定値 (パターン1)	設定値 (パターン2)
	対象リソース	exec-vm	―
	異常検出時	“failover-vm” グループのフェイルオーバ	“vm1”リソースのリトライ(3回)後フェイルオーバ(1回)

*1: 「ホスト名」は原則として FQDN 形式からドメイン名を除いたショートネームのことを指します。

*2: WebManager に接続するフローティング IP を 用意します。この IP により、障害発生時も Web ブラウザから動作している方のサーバが実行する WebManager にアクセスできます。

*3: 網掛けした項目が仮想マシンに関する設定項目です。

*4: X 3.0 以降の利用時は”virtual machine resource”を設定すると自動的に作成されます。

ホストOS間クラスタを構築する

(1) 『CLUSTERPRO X インストール&設定ガイド』[1]に従い、各ホスト OS に CLUSTERPRO X をインストールしてください。インストールする RPM は x86_64 版になります。
また、ユーザ空間モニタリソース及びシャットダウンストール監視を利用する場合は、監視方法を「keepalive」に設定してください。

(2) 『CLUSTERPRO X インストール&設定ガイド』に従い、「ホスト OS 間クラスタ設定例 (30ページ)」の例を参考にクラスタを構築します。

(3) (2)の手順の中で、仮想マシンリソースについては、本書の「仮想マシン制御用リソースの設定 (53ページ)」を参考に設定します。

(4) (2)の手順の中で、仮想マシンモニタについては、本書の「仮想マシン監視用モニタの設定(62ページ)」を参考に設定します。

(5) その他に必要な設定があれば追加します。

(6) 設定を反映します。Builder のメニューの [ファイル] から［設定の反映］または [情報ファイルのアップロード] を選択し、構成情報をアップロードしてください。

(7) クラスタを起動させる前に、フェイルオーバ対象の仮想マシンの電源を OFF にしてください。さらに、vCenter を利用しない構成の場合、フェイルオーバ対象とする仮想マシンが存在する VMware ESX の Service Console 上で下記のコマンドを実行し、仮想マシンを unregister してください。

```
# vmware-cmd -s unregister vm_path
```

(8) CLUSTERPRO を起動し、仮想マシンが正常に起動することを確認してください。

(9) 以上でクラスタの構築は終了です。

[1] http://jpn.nec.com/clusterpro/ から入手可能です。

ホストOS間クラスタの動作を確認する

(1) WebManager または clpcl コマンドでクラスタを起動してください。

(2) WebManager または clpgrp コマンドで仮想マシングループを起動してください。グループが起動しているサーバで、ゲスト OS が動作していることを確認してください。

(3) WebManager または clpgrp コマンドで仮想マシングループを移動してください。グループの移動先のサーバで、ゲスト OS が動作していることを確認してください。

(4) WebManager または clpgrp コマンドで仮想マシングループのライブマイグレーションを実行してください。グループの移動先のサーバで、ゲスト OS が動作していることを確認してください。

(5) WebManager または clpdown コマンドで、仮想マシングループが起動している物理サーバのシャットダウンまたはリブートを行ってください。この時、グループが他のサーバへ移動し、ゲスト OS が動作していることを確認してください。

(6) ゲスト OS をシャットダウンすると、仮想マシン監視用モニタが異常を検出し、回復対象の再活性またはフェイルオーバを行うことを確認してください。また、フェイルオーバ後にゲスト OS が再起動されていることを確認してください。

(7) CLUSTERPRO 以外から物理サーバをシャットダウンした場合に、他方のサーバで相手サーバの停止を検出し、仮想マシングループを起動され、ゲスト OS が再起動されることを確認してください。

(8) 上記に加え、『CLUSTERPRO X インストール & 設定ガイド 第 8 章 動作チェックを行う 動作確認テストを行う』に記載されている項目を適宜実施してください。

ホストOS間クラスタでゲスト－ホスト連携を利用する

(1) 『ホスト OS 間クラスタを構築する(34ページ)』と同様にホスト OS 間クラスタを構築してください。

(2) 『CLUSTERPRO X SingleServerSafe インストール & 設定ガイド』[1]に従い、ゲスト上に CLUSTERPRO X SingleServerSafe をインストールしてください。

(3) CLUSTERPRO Builder でゲスト側のクラスタ構成情報を編集します。スクリプトの内容は付録を参照してください。

A) 監視対象のモニタリソース (pid モニタリソース、appli モニタリソース、oracle モニタリソースなど) を追加します。

B) モニタリソースの “最終動作前スクリプトを実行する” を有効にして、[設定]ボタンを選択します。

C) ゲスト OS が Linux の場合、[置換] ボタンで preaction.sh の内容をvmpreaction.shで置き換えます。
ゲスト OS が Windows の場合、[置換] ボタンで preaction.bat の内容をvmpreaction.batで置き換えます。

D) その他の設定は適宜行ってください。

(4) CLUSTERPRO Builder で作成した構成情報をアップロードします。Builder のメニューの [ファイル] から［設定の反映］または [情報ファイルのアップロード] を選択し、構成情報をアップロードしてください。

[1] http://jpn.nec.com/clusterpro/sss/index.html から入手可能です。

ゲストOS間クラスタの構築

ゲストOS間クラスタを構築する

(1) 仮想マシンを作成していない場合は、「仮想マシンの作成 (25ページ)」を参考に仮想マシンを作成してください。

(2) CLUSTERPRO がサポートするゲスト OS を仮想マシンにインストールしてください。

(3) 『CLUSTERPRO X インストール & 設定ガイド』に従い、ゲスト OS に CLUSTERPRO をインストールしてください。

(4) 『CLUSTERPRO X インストール & 設定ガイド』に従い、CLUSTERPRO Builder でクラスタを構築してください。

(5) 設定を反映します。Builder のメニューの [ファイル] から［設定の反映］または [情報ファイルのアップロード] を選択し、構成情報をアップロードしてください。

ゲストOS間クラスタの動作を確認する

(1) WebManager または clpcl コマンドでクラスタを起動してください。

(2) WebManager または clpgrp コマンドでフェイルオーバグループを移動してください。フェイルオーバグループの移動先のサーバで、フェイルオーバグループが起動していることを WebManager または clpstat コマンドで確認してください。

(3) WebManager または clpdown コマンドで、フェイルオーバグループが起動している仮想マシンのシャットダウンまたはリブートを行ってください。この時、フェイルオーバグループが他のサーバで起動していることを WebManager または clpstat コマンドで確認してください。

(4) CLUSTERPRO 以外から物理サーバの電源を落とした場合に、他方のサーバで相手サーバの停止を検出し、フェイルオーバグループを起動していることを WebManager または clpstat コマンドで確認してください。

(5) 上記に加え、『CLUSTERPRO X インストール & 設定ガイド 第 8 章 動作チェックを行う 動作確認テストを行う』に記載されている項目を適宜実施してください。

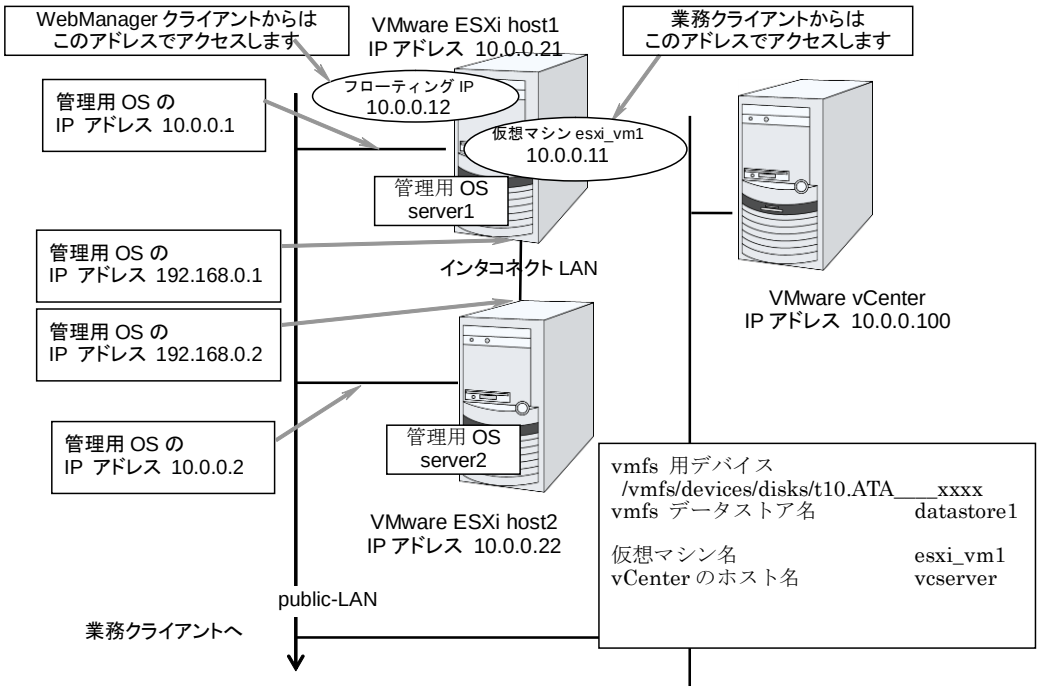
管理用OS間クラスタの構築

管理用OS間クラスタ設定例

管理用 OS 間クラスタは CLUSTERPRO X 3.1 以降でのみ使用可能です。

表 4 ： 管理用 OS 間クラスタの構築対応表

仮想マシンの制御方法	CLUSTERPRO X 3.1 for Linux
EXEC リソースとカスタムモニタリソースを使用する	×
仮想マシンリソースと仮想マシンモニタリソースを使用する	○



2 ノードの構成設定例

設定対象	設定パラメータ	設定値
クラスタ構成	クラスタ名	cluster
	サーバ数	2
	フェイルオーバ グループ数	2
	モニタ リソース数	5
ハートビート リソース	LAN ハートビート数	2
1 台目のサーバの情報 (マスタ サーバ)	サーバ名*1	server1
	インタコネクトの IP アドレス (専用)	192.168.0.1

設定対象	設定パラメータ	設定値
	インタコネクトの IP アドレス (バックアップ)	10.0.0.1
2 台目のサーバの情報	サーバ名*1	server2
	インタコネクトの IP アドレス (専用)	192.168.0.2
	インタコネクトの IP アドレス (バックアップ)	10.0.0.2
管理用のグループ	タイプ	フェイルオーバ
(WebManager 用)	グループ名	ManagementGroup
	起動サーバ	全てのサーバ
	グループ リソース数	1
管理用グループのグループリソース *2	タイプ	floating ip resource
	グループ リソース名	ManagementIP
	IPアドレス	10.0.0.12
業務用のグループ	タイプ	仮想マシン
	グループ名	virtualmachine1
	起動サーバ	全てのサーバ
	グループ リソース数	1
1 つ目のグループリソース	タイプ	virtual machine resource
（仮想マシン制御用リソース）*3	グループ リソース名	vm1
	仮想マシンの種類	vSphere
	クラスタサービスインストール先	ゲスト
	仮想マシン名	esxi_vm1
	データストア名	datastore1
	ホストのIPアドレス(共通)	10.0.0.21
	ホストのIPアドレス (サーバ個別設定：server1)	10.0.0.21
	ホストのIPアドレス (サーバ個別設定：server2)	10.0.0.22
	vCenterを使用する	オン
	vCenterのホスト名	vcserver
	リクエストタイムアウト	120
	仮想マシン起動待ち時間	30
1 つ目のモニタリソース	タイプ	user mode monitor
(デフォルト作成)	モニタ リソース名	userw
2 つ目のモニタリソース	タイプ	custom monitor
(ホスト：ディスク監視用モニタ) *4	モニタ リソース名	genw_disk
	スクリプト	clphostmon_wrap.sh

設定対象	設定パラメータ	設定値
	監視タイミング	常時
	clphostmon.plに設定する監視対象	/vmfs/devices/disks/t10.ATA____xxxx
	回復動作	最終動作のみ実行
	回復スクリプト	stdnhost_wrap.sh
	最終動作前にスクリプトを実行する	オン
3 つ目のモニタリソース	タイプ	custom monitor
(ホスト:NIC監視用モニタ) *4	モニタ リソース名	genw_nics1
	スクリプト	clphostmon_wrap.sh
	監視タイミング	常時
	clphostmon.plに設定する監視対象	vmnic0 (Public LANの仮想スイッチインタフェース)
	回復動作	回復対象に対してフェイルオーバ実行
	異常検出時	“ManagementGroup” グループのフェイルオーバ
4 つ目のモニタリソース	タイプ	custom monitor
(ホスト:NIC監視用モニタ) *4	モニタ リソース名	genw_nics2
	スクリプト	clphostmon_wrap.sh
	監視タイミング	常時
	clphostmon.plに設定する監視対象	vmnic0 (Public LANの仮想スイッチインタフェース)
	回復動作	回復対象に対してフェイルオーバ実行
	フェイルオーバ実行前にマイグレーションを実行する	オン
	異常検出時	“virtualmachine1” グループのフェイルオーバ
5 つ目のモニタリソース	タイプ	virtual machine monitor
（仮想マシン監視用モニタ）*3*5	モニタ リソース名	vmw1
	仮想マシンリソース	vm1
	外部マイグレーション発生時の待ち時間	15(秒)
	監視タイミング	常時
	異常検出時	“vm1”リソースのリトライ(3回)後フェイルオーバ(1回)

*1:「ホスト名」は原則として FQDN 形式からドメイン名を除いたショートネームのことを指します。

*2: WebManager に接続するフローティング IP を 用意します。この IP により、障害発生時も Web ブラウザから動作している方のサーバが実行する WebManager にアクセスできます。

*3: 網掛けした項目が仮想マシンに関する設定項目です。

*4: vMA を使用している場合にのみ設定できるホスト監視に関する項目です。

*5:”virtual machine resource”を設定すると自動的に作成されます。

管理用OS間クラスタを構築する

(1) 『CLUSTERPRO X インストール&設定ガイド』[1]に従い、各管理用 OS に CLUSTERPRO X 3.1 for Linux をインストールしてください。vMA を使用している場合は、インストールする RPM は x86_64 版になります。

(2) 『CLUSTERPRO X インストール&設定ガイド』に従い、「管理用 OS 間クラスタ設定例(39ページ)」の例を参考にクラスタを構築します。

(3) (2)の手順の中で、仮想マシンリソースについては、本書の「管理用 OS で仮想マシンリソースを利用する場合(X3.1 以降)(59ページ)」を参考に設定します。

(4) (2)の手順の中で、仮想マシンモニタについては、本書の「仮想マシン監視用モニタの設定(62ページ)」を参考に設定します。

(5) (2)の手順の中で、ホスト監視用モニタについては、本書の「ホスト監視用モニタの設定(45ページ)」を参考に設定します。

(6) その他に必要な設定があれば追加します。

(7) 設定を反映します。Builder のメニューの [ファイル] から［設定の反映］または [情報ファイルのアップロード] を選択し、構成情報をアップロードしてください。

(8) CLUSTERPRO を起動し、仮想マシンが正常に起動することを確認してください。

(9) 以上でクラスタの構築は終了です。

[1] http://jpn.nec.com/clusterpro/ から入手可能です。

管理用OS間クラスタの動作を確認する

(1) WebManager または clpcl コマンドでクラスタを起動してください。

(2) WebManager または clpgrp コマンドで仮想マシングループを起動してください。グループが起動しているサーバで、ゲスト OS が動作していることを確認してください。

(3) WebManager または clpgrp コマンドで仮想マシングループを移動してください。グループの移動先のサーバで、ゲスト OS が動作していることを確認してください。

(4) WebManager または clpgrp コマンドで仮想マシングループのライブマイグレーションを実行してください。グループの移動先のサーバで、ゲスト OS が動作していることを確認してください。

(5) ゲスト OS をシャットダウンすると、仮想マシン監視用モニタが異常を検出し、回復対象の再活性またはフェイルオーバを行うことを確認してください。また、フェイルオーバ後にゲスト OS が再起動されていることを確認してください。

(6) VMware vCenter から物理サーバをシャットダウンした場合に、他方のサーバで相手サーバの停止を検出し、仮想マシングループを起動され、ゲスト OS が再起動されることを確認してください。

(7) 上記に加え、『CLUSTERPRO X インストール & 設定ガイド 第 8 章 動作チェックを行う 動作確認テストを行う』に記載されている項目を適宜実施してください。

ホスト監視用モニタの設定

本機能を使用する場合は、管理用OSにvMAを使用している必要があります。

(1) 各 vMA のコンソールにて以下のコマンドを実行し、監視対象の VMware ESXi を登録します。

```
# vifp addserver VMware ESXi's hostname
root@esxi_host1's password: login password for VMware ESXi
```

以下のコマンドを実行し、正常に登録されていることを確認してください。

```
# vifp listservers
esxi_host1 ESXi
```

(2) 対象の VMware ESXi よりハードウェアの情報を取得します。
以下のコマンドを実行することで、以降のコマンド実行対象に対象の VMware ESXi を指定します。

```
# vifptarget --set registerd VMware ESXi's hostname
```

以下のコマンドを実行し、ストレージの情報を取得します。一覧表示されたストレージより、監視したいストレージのデバイスファイルのパス[Devfs Path]を控えてください。

```
#[esxi_host1]# vicfg-scsidevs --list
t10.ATA_____xxxx
   Device Type: disk
   Size: 238419 MB
   Display Name: Local ATA Disk
   Plugin: NMP
   Console Device: /vmfs/devices/disks/t10.ATA_____xxxx
   Devfs Path: /vmfs/devices/disks/t10.ATA_____xxxx
   Vendor: ATA      Model: WDC WD2502ABYS-0  Revis: 02.0
   SCSI Level: 5  Is Pseudo:   Status:
   Is RDM Capable:  Is Removable:
   ……
```

以下のコマンドを実行し、ネットワークアダプタの情報を取得します。監視したいネットワークアダプタのデバイス名[Name]を控えてください。

```
#[esxi_host1]# vicfg-nics --list
Name     PCI       Driver Link   Speed     ……
vmnic0   01:00.0   igb    Up     100Mbps   ……
vmnic1   01:00.1   igb    Up     1000Mbps  ……
```

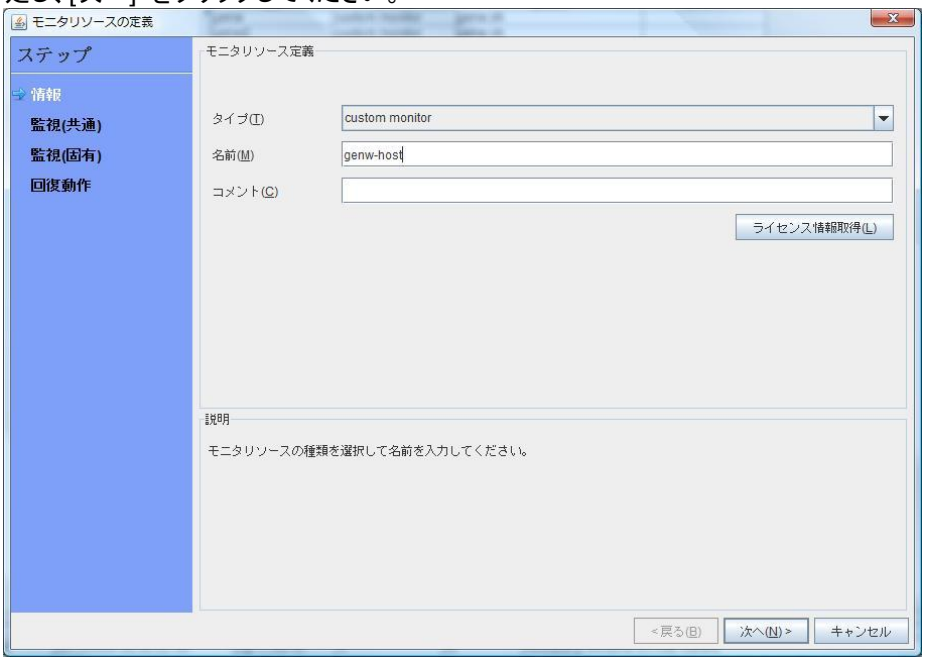
最後に以下のコマンドを実行し、コマンド実行対象より VMware ESXi を解除します。

```
# vifptarget -clear
```

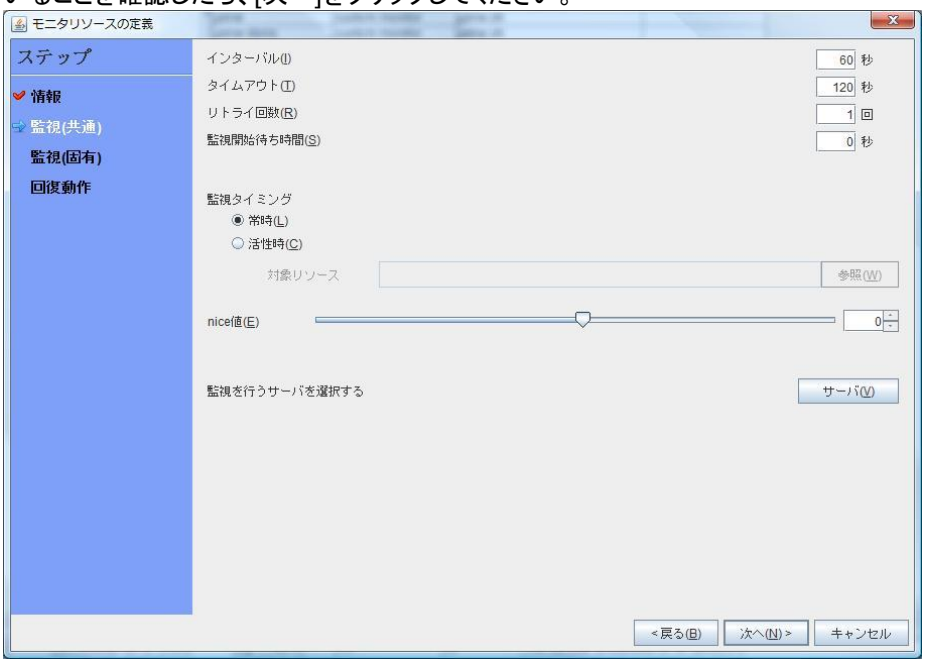
(3) 前述の手順で得た情報を元に本書のサンプルスクリプト clphostmon.pl(84ページ)を編集してください。

(4) CLUSTERPRO Builder の画面で、ツリービューの [Monitors] を右クリックし、[モニタリソースの追加] をクリックしてください。

(5) 仮想マシン監視用カスタムモニタリソースを登録します。[タイプ] から [custom monitor resource] を選択してください。カスタムモニタリソースに任意の名前 (genw-host) を設定し、[次へ] をクリックしてください。

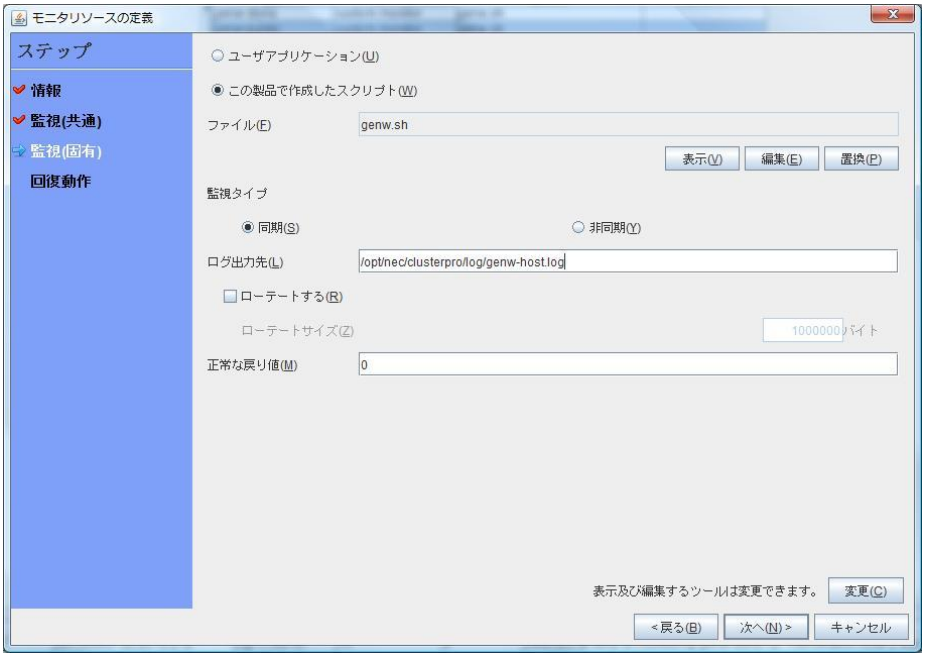


(6) [監視]の設定を行います。ホストへの接続エラー(ビジーなど)による誤検知を避けるため、[リトライ回数]は 1 回以上に設定することを推奨します。[監視タイミング]が[常時]になっていることを確認したら、[次へ]をクリックしてください。



(7) ホスト監視のためのスクリプトを設定します。本書のサンプルスクリプト clphostmon.pl (84ページ)を各 vMA の同一のパスに格納してください。
(*/opt/nec/clusterpro/work/clphostmon/clphostmon.pl* など)

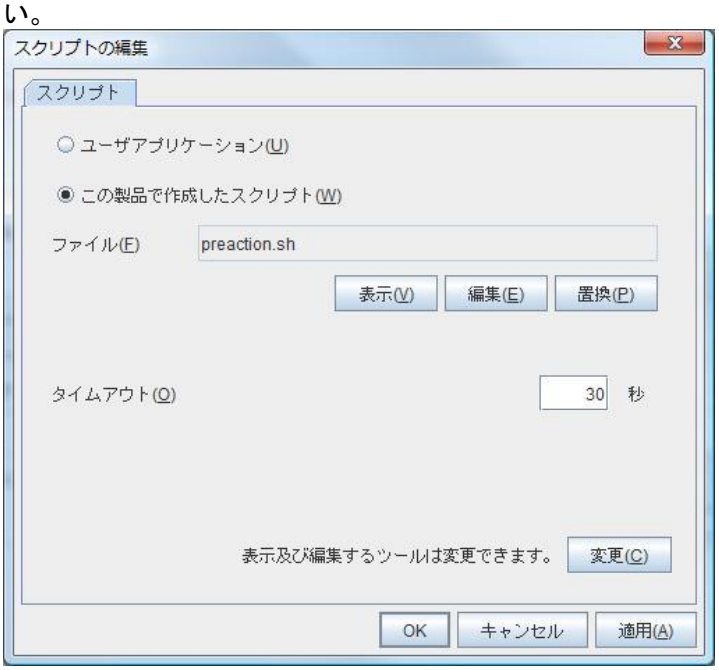
(8) [置換] をクリックし、genw.shを本書のサンプルスクリプト clphostmon_wrap.sh (83ページ)で置換し、[監視タイプ] が[同期]になっていることを確認したら [次へ] をクリックしてください。



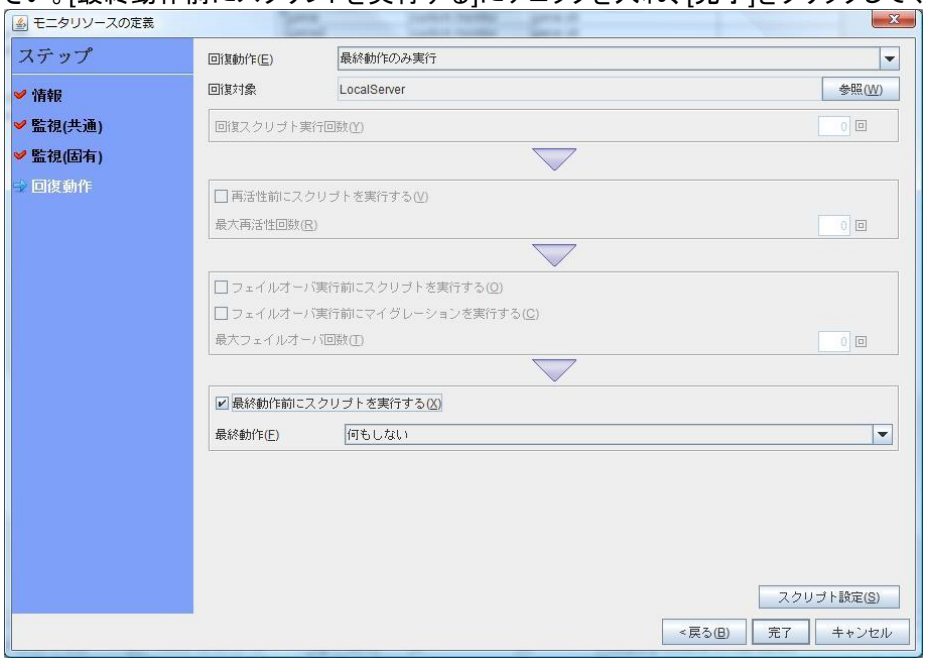
(9) [回復動作]を設定します。管理用 OS 間クラスタ設定例(39ページ)の 2 つ目のモニタリソース(ホスト:ディスク監視用モニタ)の設定手順は(9) ～ (12)を参照してください。
3 つ目のモニタリソース(ホスト:NIC 監視用モニタ)の設定手順は(13)を参照してください。
4 つ目のモニタリソース(ホスト:NIC 監視用モニタ)の設定手順は(14)を参照してください。

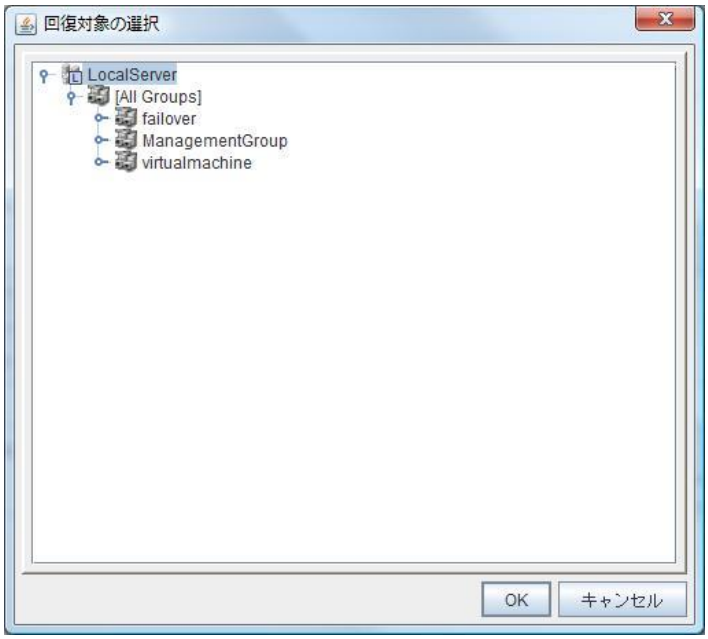
(10) 2 つ目のモニタリソース用の[回復動作]の設定手順を説明します。異常検知時に管理用 OS が所属するホストをシャットダウンするよう[回復動作]を設定します。本書のサンプルスクリプトshutdownhost.pl(90ページ)を各 vMA の同一のパスに格納してください。
(*/opt/nec/clusterpro/work/shutdownhost/shutdownhost.pl* など)

(11) [スクリプト設定]をクリックし、[スクリプトの編集]を開いてください。[置換]をクリックし、preaction.sh を本書のサンプルスクリプトstdnhost_wrap.sh(89ページ)で置換してください。

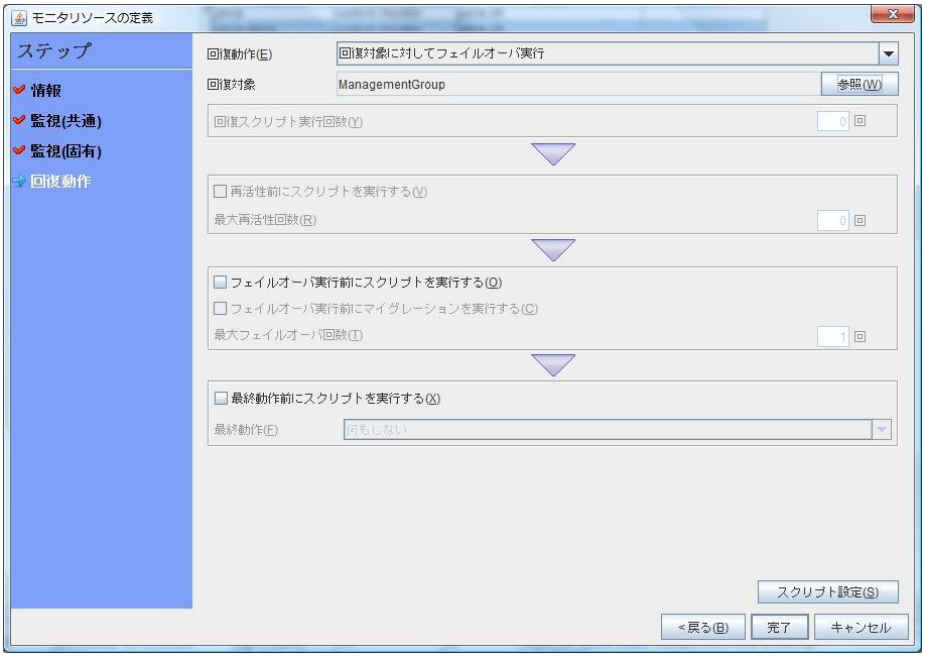


(12) [回復動作]を[最終動作のみ実行]に設定し、 [回復対象]に[LocalServer]を選択してください。[最終動作前にスクリプトを実行する]にチェックを入れ、[完了]をクリックしてください。

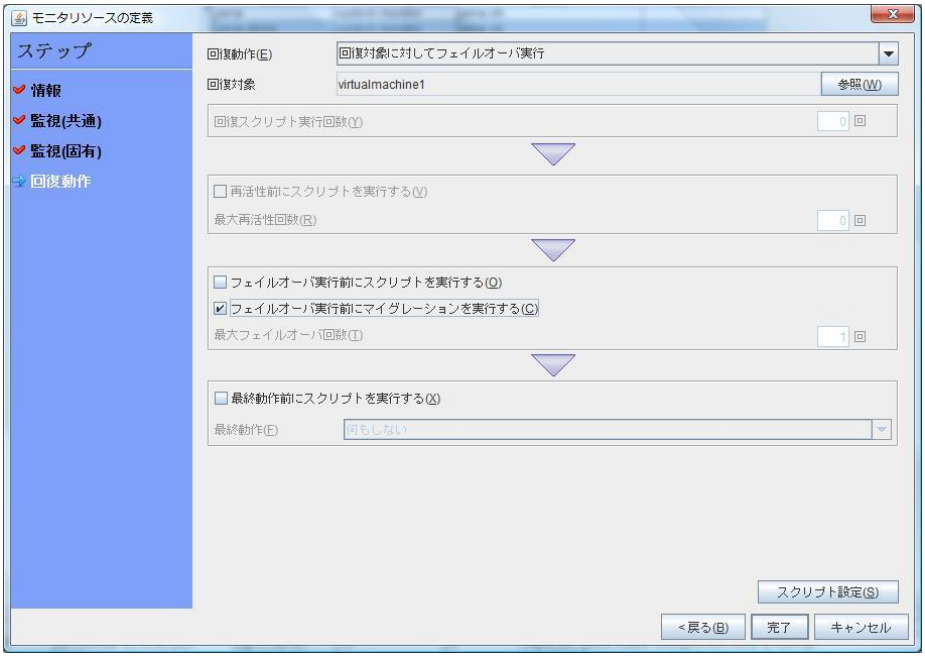




(13) 3 つ目のモニタリソース用の[回復動作]の設定手順を説明します。[回復動作]に[回復対象に対してフェイルオーバ実行]を、[回復対象]に[ManagementGroup]を選択し、[完了]をクリックしてください。



(14) 4 つ目のモニタリソース用の[回復動作]の設定手順を説明します。[回復動作]に[回復対象に対してフェイルオーバ実行]を、[回復対象]に[virtualmachine1]を選択してください。次に[フェイルオーバ実行前にマイグレーションを実行する]にチェックを入れ、[完了]をクリックしてください。



物理サーバー仮想マシン間クラスタの構築

物理サーバー仮想マシン間クラスタを構築する

(1) 仮想マシンを作成していない場合は、「仮想マシンの作成 (25ページ)」を参考に仮想マシンを作成してください。

(2) CLUSTERPRO がサポートするゲスト OS を仮想マシンにインストールしてください。

(3) 『CLUSTERPRO X インストール & 設定ガイド』に従い、物理サーバ OS 及びゲスト OS に CLUSTERPRO をインストールしてください。

(4) 『CLUSTERPRO X インストール & 設定ガイド』に従い、CLUSTERPRO Builder でクラスタを構築してください。

(5) 設定を反映します。Builder のメニューの [ファイル] から［設定の反映］または [情報ファイルのアップロード] を選択し、構成情報をアップロードしてください。

物理サーバー仮想マシン間クラスタの動作を確認する

(1) WebManager または clpcl コマンドでクラスタを起動してください。

(2) WebManager または clpgrp コマンドでフェイルオーバグループを移動してください。フェイルオーバグループの移動先のサーバで、フェイルオーバグループが起動していることを WebManager または clpstat コマンドで確認してください。

(3) WebManager または clpdown コマンドで、フェイルオーバグループが起動しているサーバのシャットダウンまたはリブートを行ってください。この時、フェイルオーバグループが他のサーバで起動していることを WebManager または clpstat コマンドで確認してください。

(4) CLUSTERPRO 以外から物理サーバの電源を落とした場合に、他方のサーバで相手サーバの停止を検出し、フェイルオーバグループを起動していることを WebManager または clpstat コマンドで確認してください。

(5) 上記に加え、『CLUSTERPRO X インストール & 設定ガイド 第 8 章 動作チェックを行う 動作確認テストを行う』に記載されている項目を適宜実施してください。

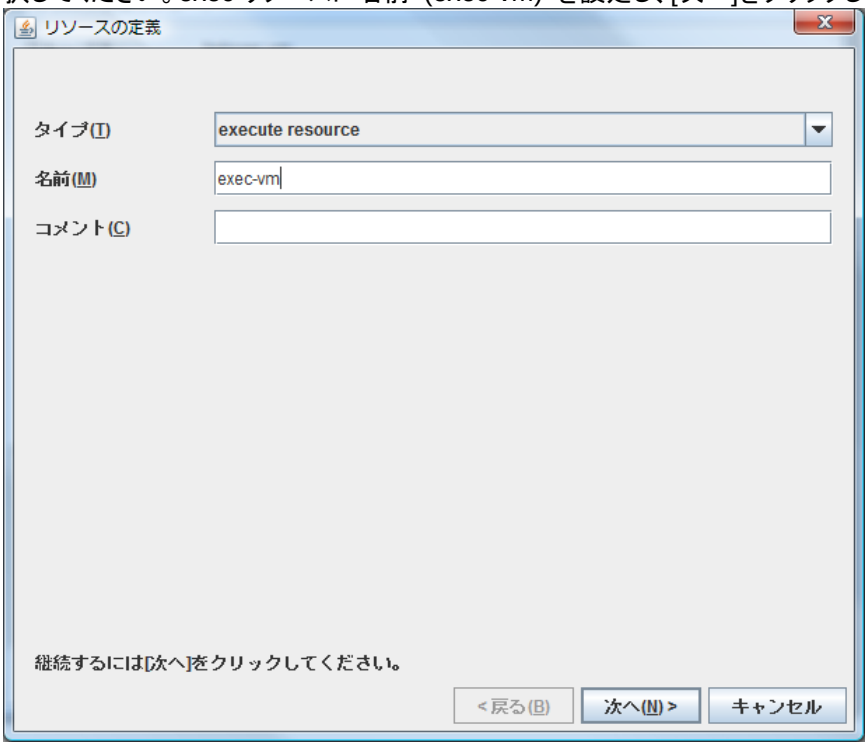
仮想マシン制御用リソースの設定

EXECリソースを利用する場合

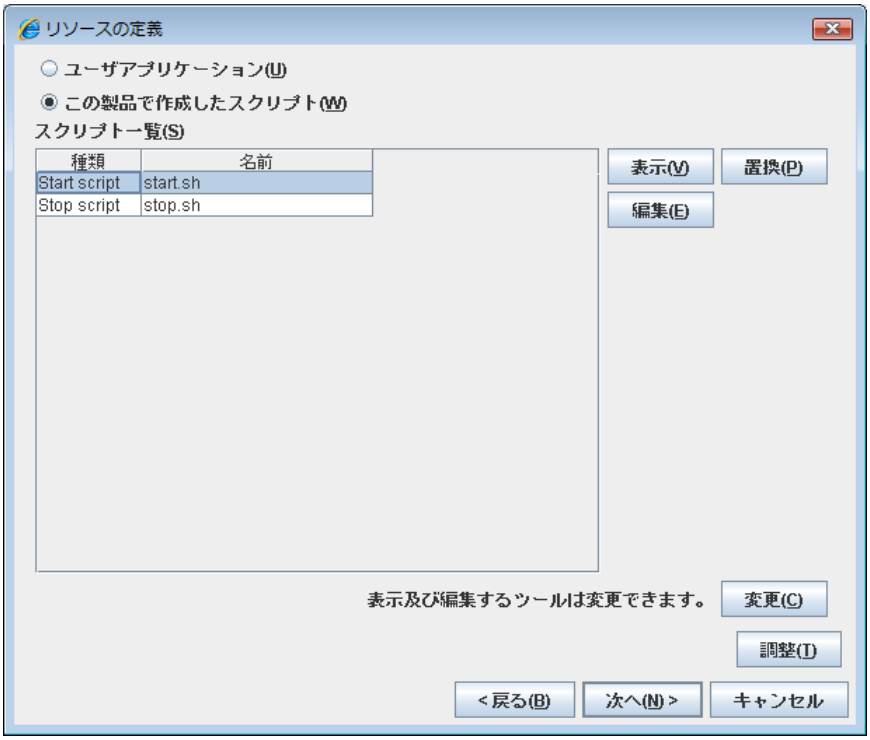
※ CLUSTERPRO X 3.0以降をご利用の場合、仮想マシンリソースの利用を推奨します。

(1) CLUSTERPRO Builder の画面で、ツリービューのフェイルオーバグループ名 (failover-vm) を右クリックし、[リソースの追加] をクリックしてください。

(2) 仮想マシン制御用 EXEC リソースを登録します。[タイプ] から [execute resource] を選択してください。exec リソースに名前 (exec-vm) を設定し、[次へ]をクリックしてください。

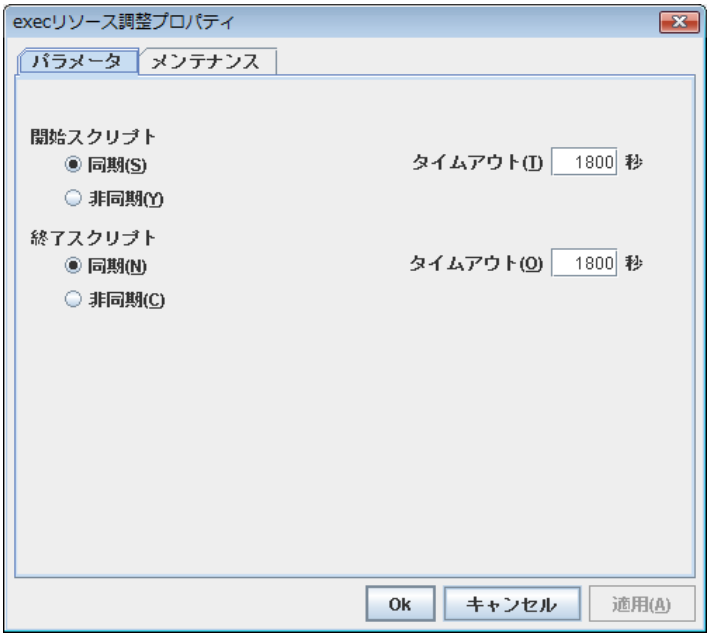


(3) [Start script] を選択し、[置換] をクリックしてください。ファイル選択画面が表示されるので、本書のサンプルスクリプト vmpower.start.pl (69ページ) を選択し、start.sh を置換してください。

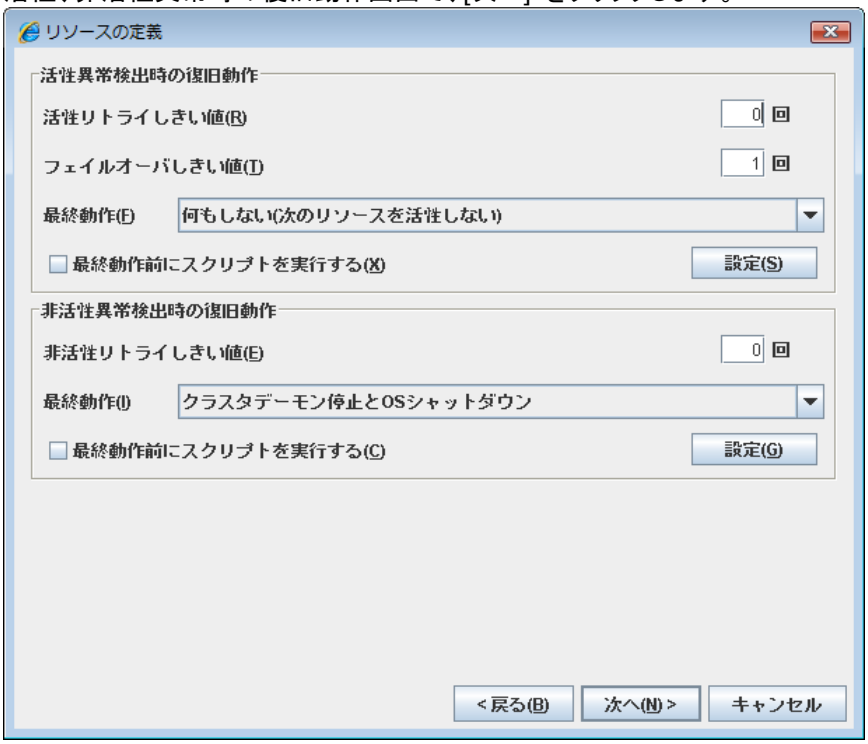


(4) [Stop script] を選択し、[置換]をクリックしてください。ファイル選択画面が表示されるので、本書のサンプルスクリプト vmpower.stop.pl (74ページ) を選択し、stop.sh を置換してください。

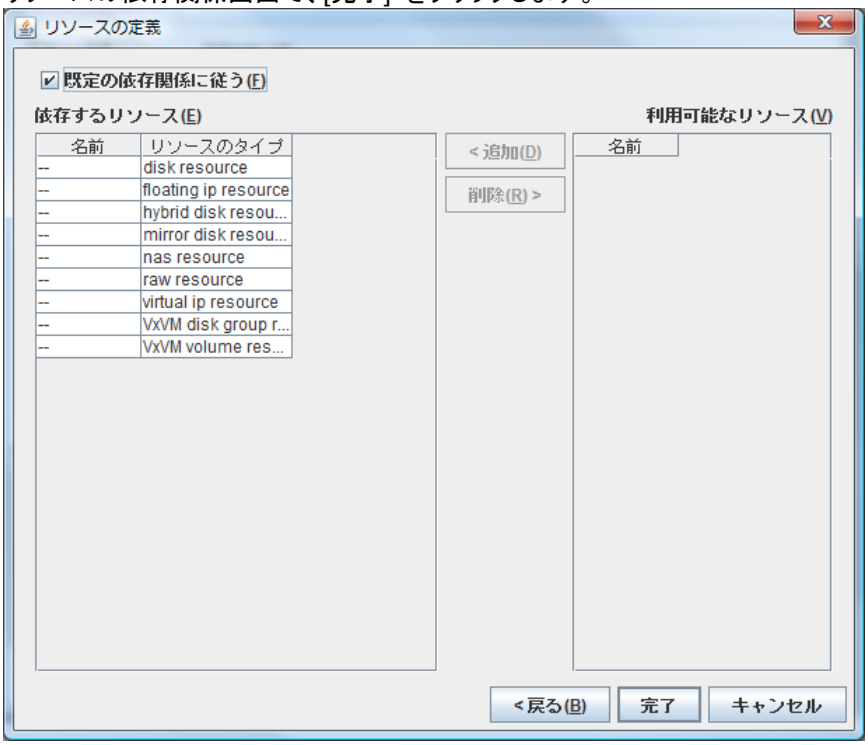
(5) [調整] をクリックし、非大域ゾーンの活性、非活性タイムアウトを適宜変更します。[次へ] をクリックします。



(6) 活性、非活性異常時の復旧動作画面で、[次へ] をクリックします。



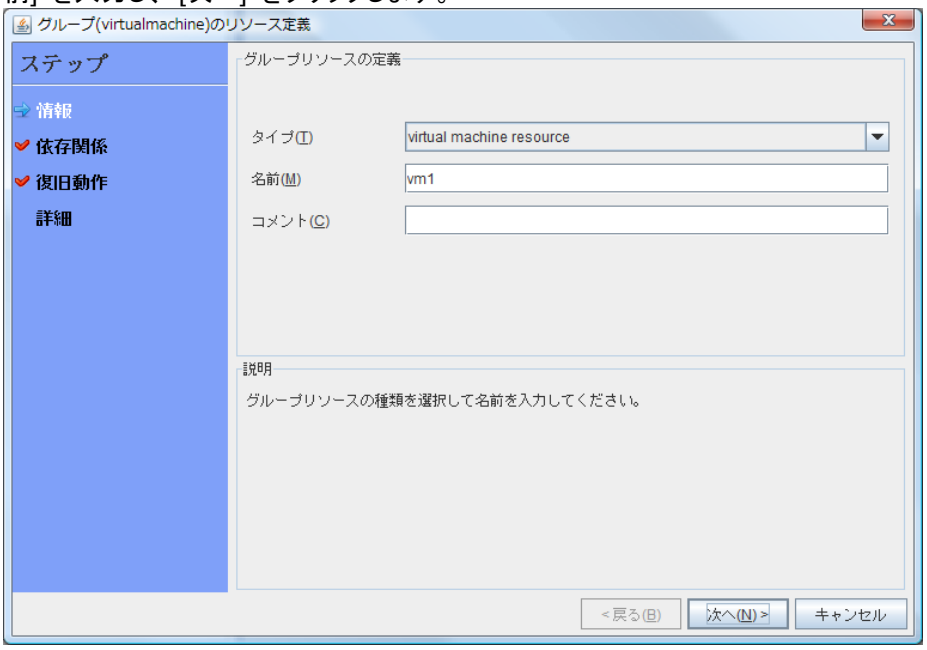
(7) リソースの依存関係画面で、[完了] をクリックします。



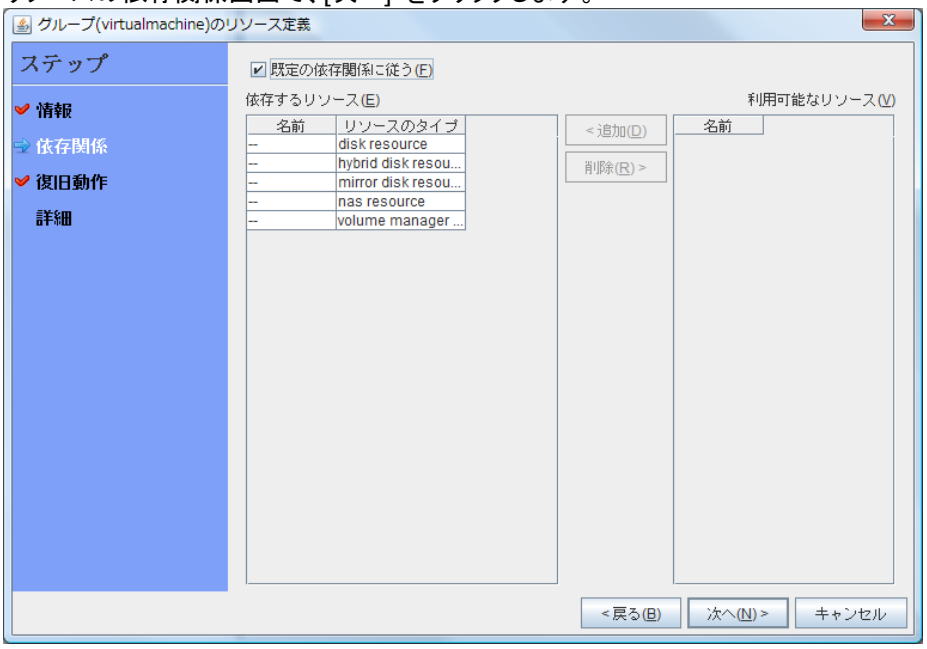
仮想マシンリソースを利用する場合(X3.0以降)

(1) CLUSTERPRO Builder の画面で、ツリービューの仮想マシングループ名(virtualmachine1) を右クリックし、[リソースの追加] をクリックしてください。

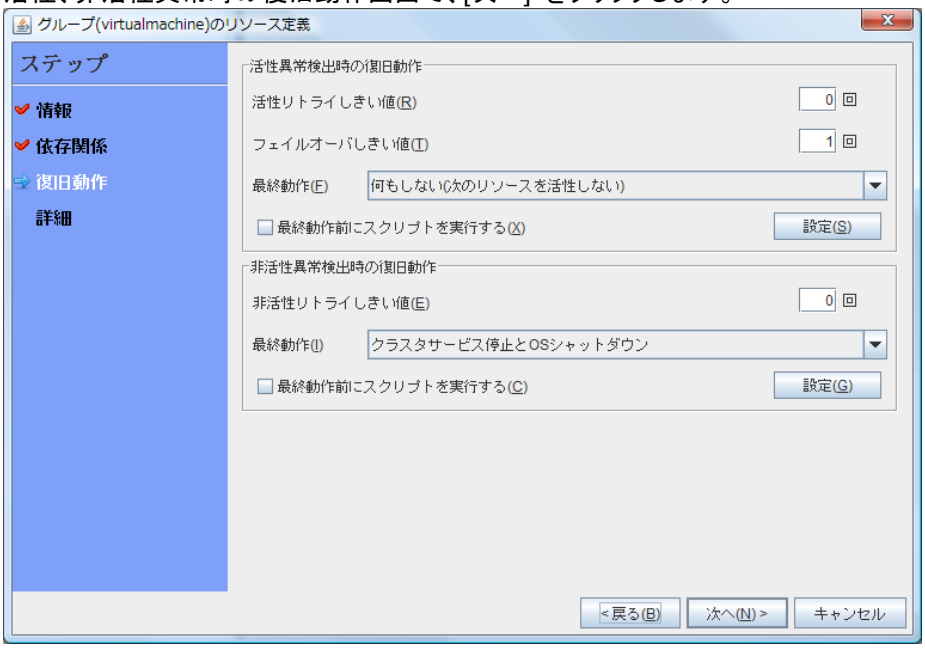
(2) 仮想マシンリソースを登録します。[タイプ] に [virtual machine resource] を選択、[名前] を入力し、 [次へ] をクリックします。



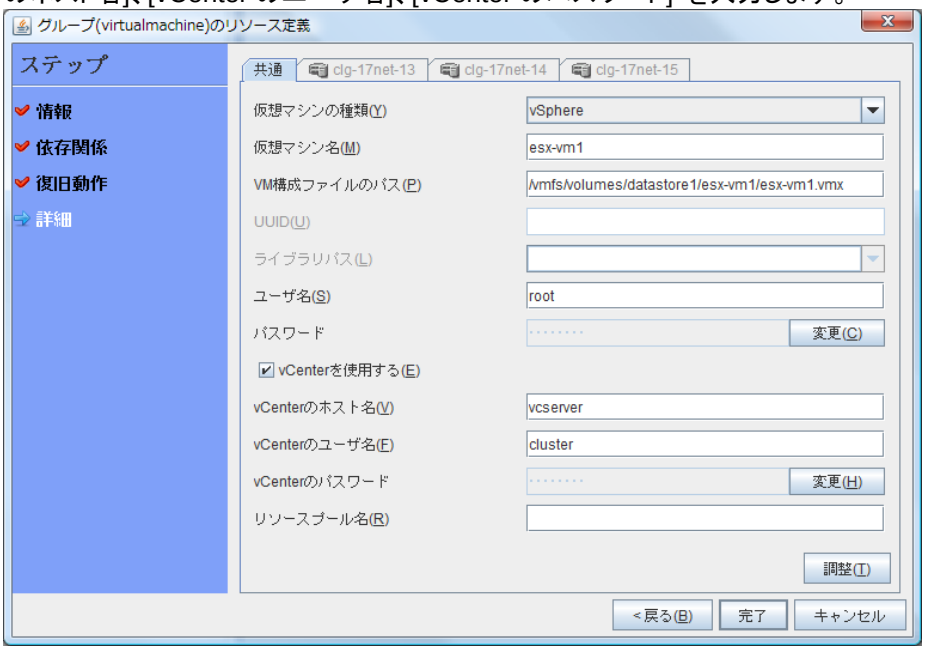
(3) リソースの依存関係画面で、[次へ] をクリックします。



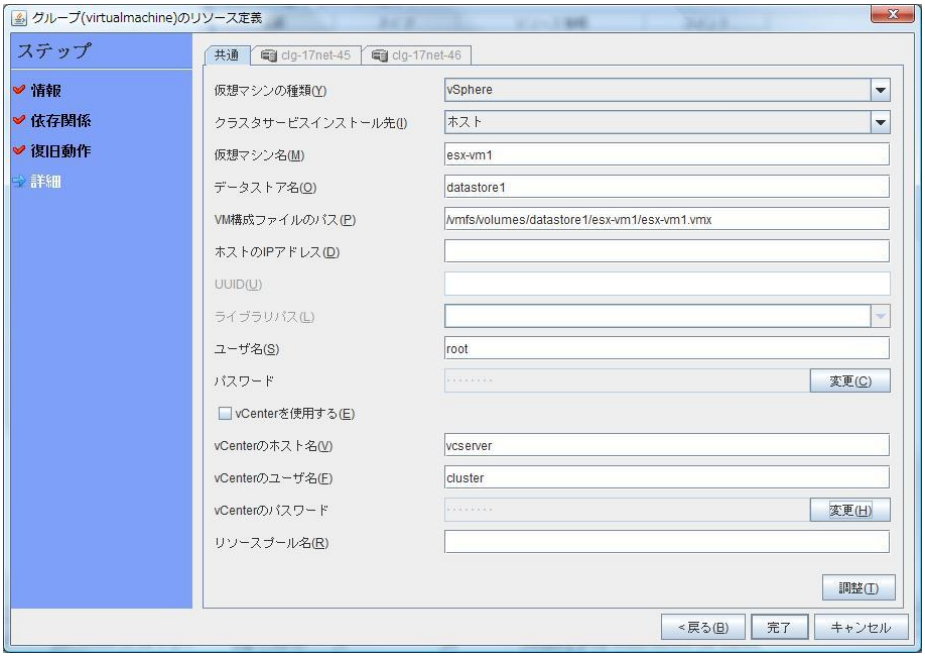
(4) 活性、非活性異常時の復旧動作画面で、[次へ] をクリックします。



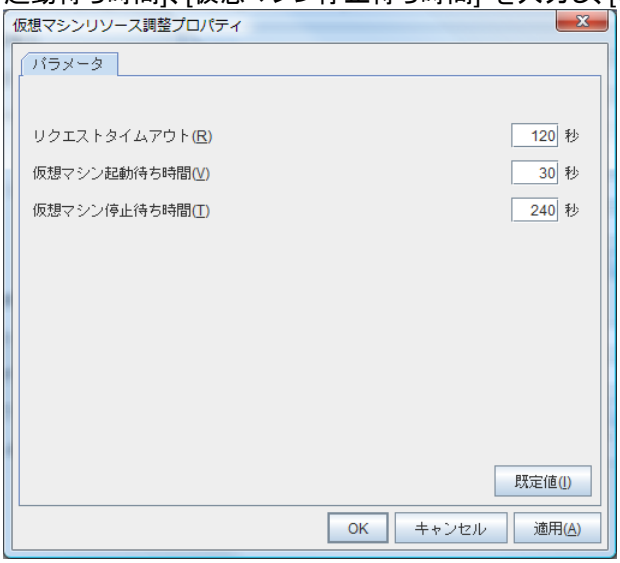
(5) [仮想マシンの種類] に [vSphere] を選択し、[仮想マシン名]、[VM 構成ファイルのパス]、[ユーザ名]、[パスワード] を設定します。
X 3.1 以降の場合は、[クラスタサービスのインストール先]に[ホスト]を選択してください。
vCenter を使用しない場合は、[vCenter を使用する] チェックボックスがオフであることを確認します。
vCenter を使用する場合は、[vCenter を使用する] チェックボックスをオンにし、[vCenter のホスト名]、[vCenter のユーザ名]、[vCenter のパスワード] を入力します。



※ X3.1 以降の画面は以下。



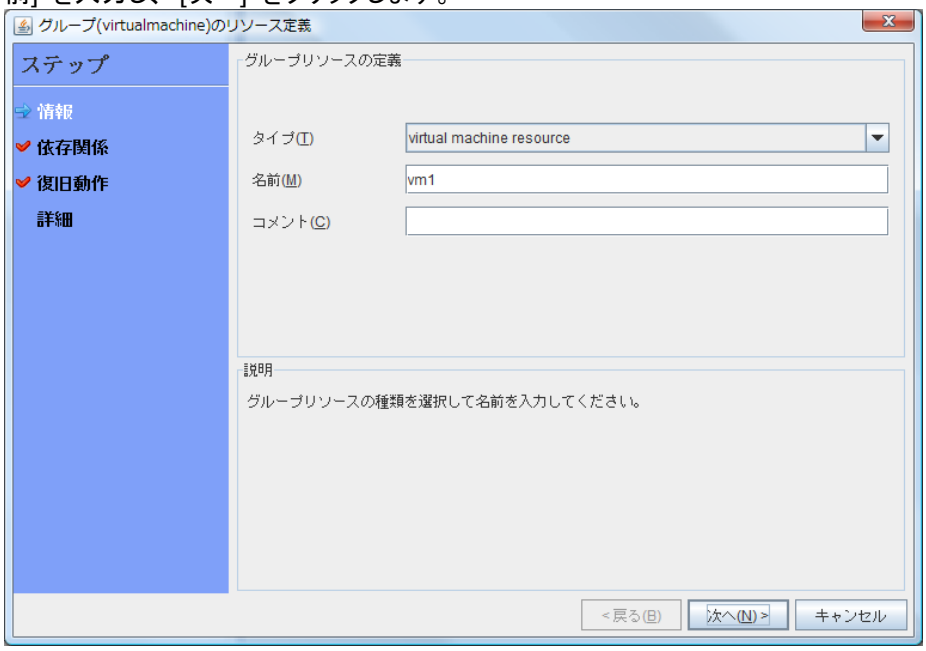
(6) [調整] ボタンをクリックして調整プロパティを表示し、[リクエストタイムアウト]、[仮想マシン起動待ち時間]、[仮想マシン停止待ち時間] を入力し、[OK] をクリックします。



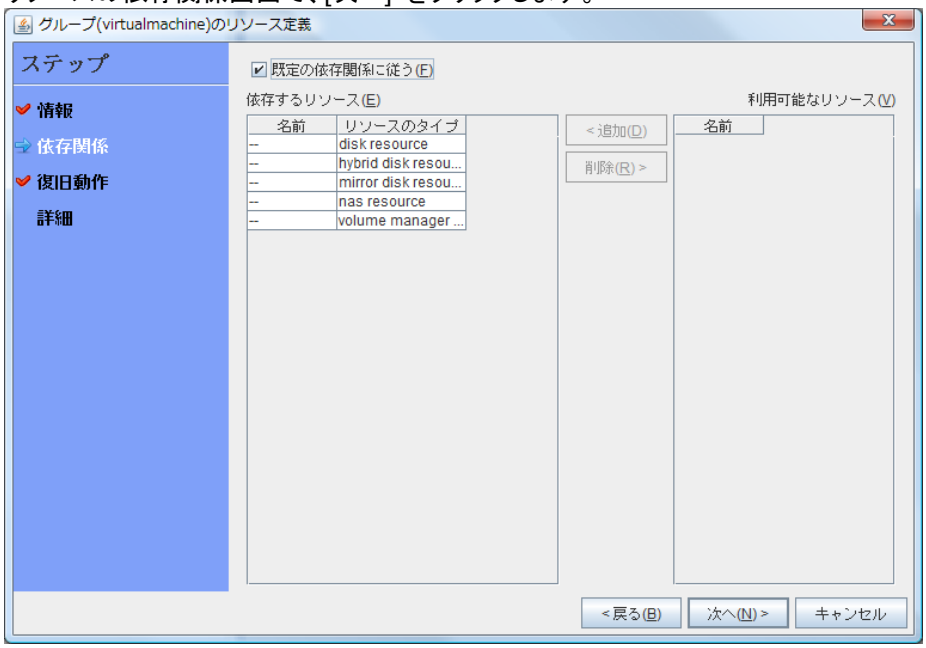
管理用OSで仮想マシンリソースを利用する場合(X3.1以降)

(1) CLUSTERPRO Builder の画面で、ツリービューの仮想マシングループ名 (virtualmachine1) を右クリックし、[リソースの追加] をクリックしてください。

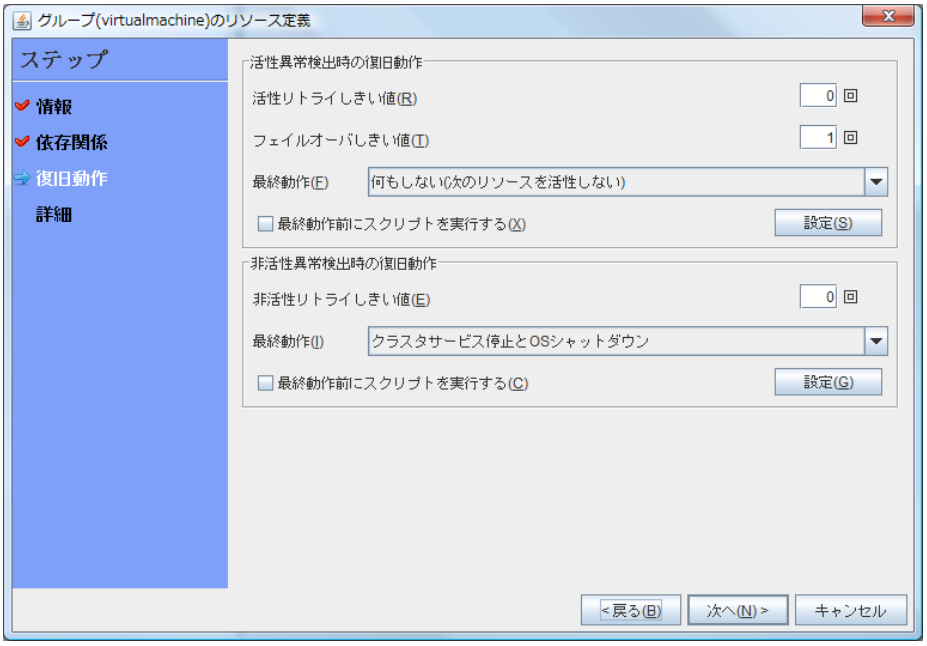
(2) 仮想マシンリソースを登録します。[タイプ] に [virtual machine resource] を選択、[名前] を入力し、 [次へ] をクリックします。



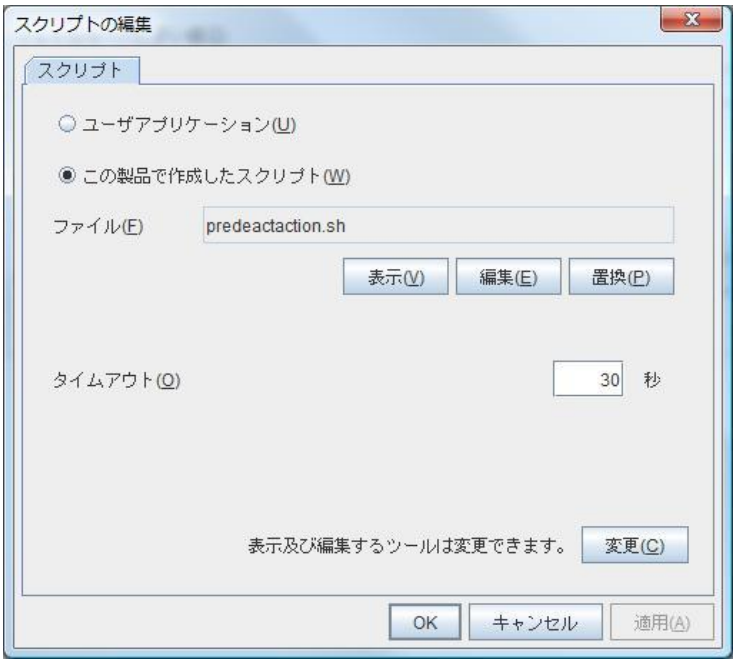
(3) リソースの依存関係画面で、[次へ] をクリックします。



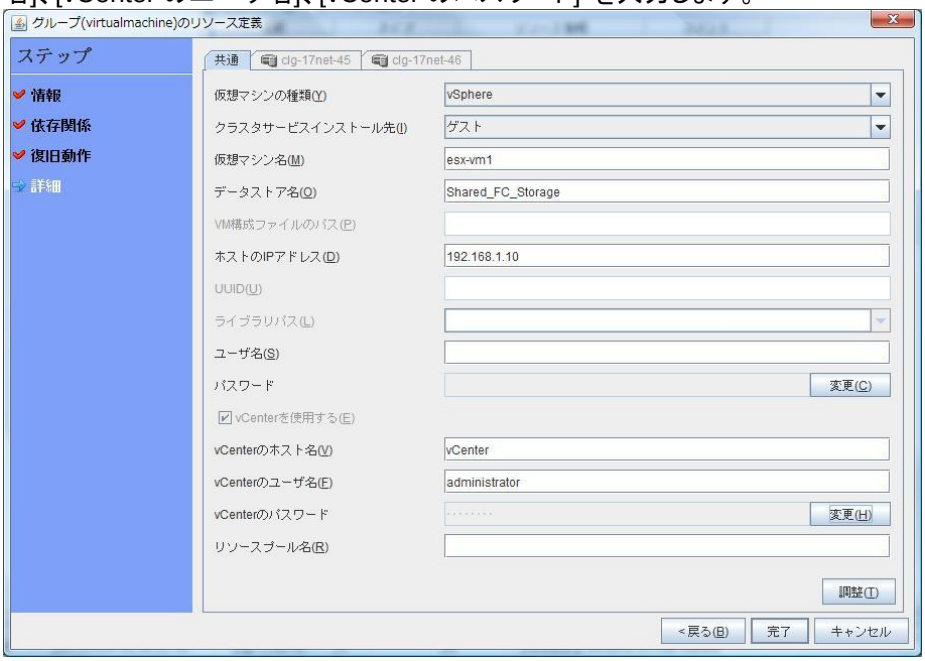
(4) 活性、非活性異常時の復旧動作画面を設定し、[次へ] をクリックします。
OS シャットダウンを選択している場合、シャットダウンされるのはホストではなく、管理用 OS になります。管理用 OS が所属するホスト(VMware ESXi)をシャットダウンしたい場合は、[最終動作前にスクリプトを実行する]にチェックを入れ、[設定]をクリックしてください。
※ ホストのシャットダウンは管理用 OS に vMA を使用している場合にのみ実行可能です。



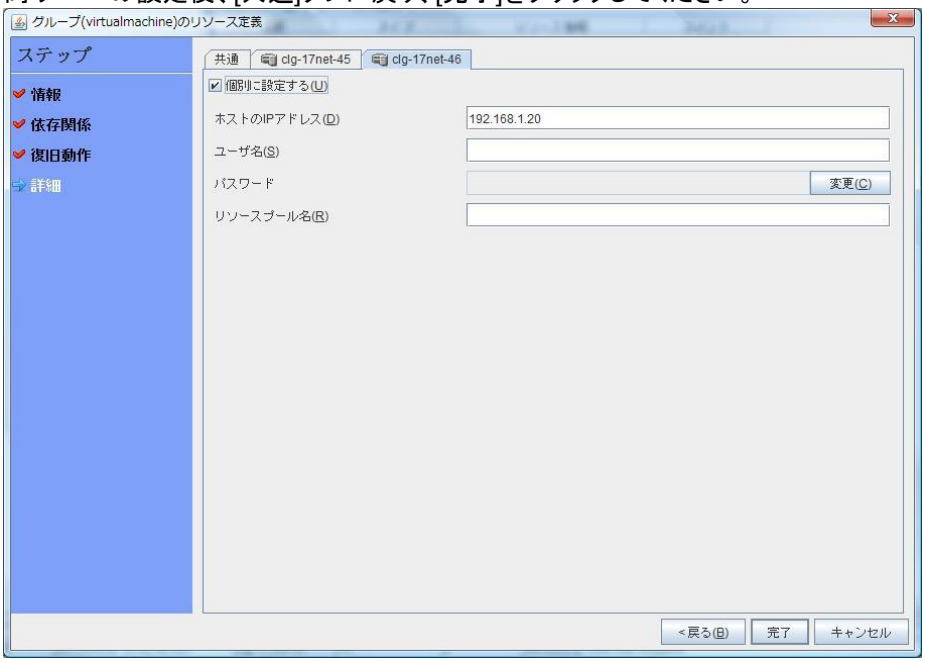
[スクリプトの編集]画面で[置換]をクリックし、preactaction.sh、predeactaction.sh を本書のサンプルスクリプトstdnhost_wrap.sh (89ページ)を置換してください。事前に各 vMA にホストの登録(45(1)を参照)を行い、shutdownhost.pl(90ページ)を同一のパスに格納しておく必要があります。(48ページの(9) ～ (11)も参照してください)



(5) [仮想マシンの種類] に [vSphere] を、[クラスタサービスインストール先]には[ゲスト]を選択します。
[共通]タブでは[仮想マシン名]、[データストア名]、[ホストの IP アドレス]、[vCenter のホスト名]、[vCenter のユーザ名]、[vCenter のパスワード] を入力します。



次に各サーバのタブを開き、それぞれが所属する VMware ESXi の[ホストの IP アドレス]を入力してください。
両サーバの設定後、[共通]タブに戻り、[完了]をクリックしてください。



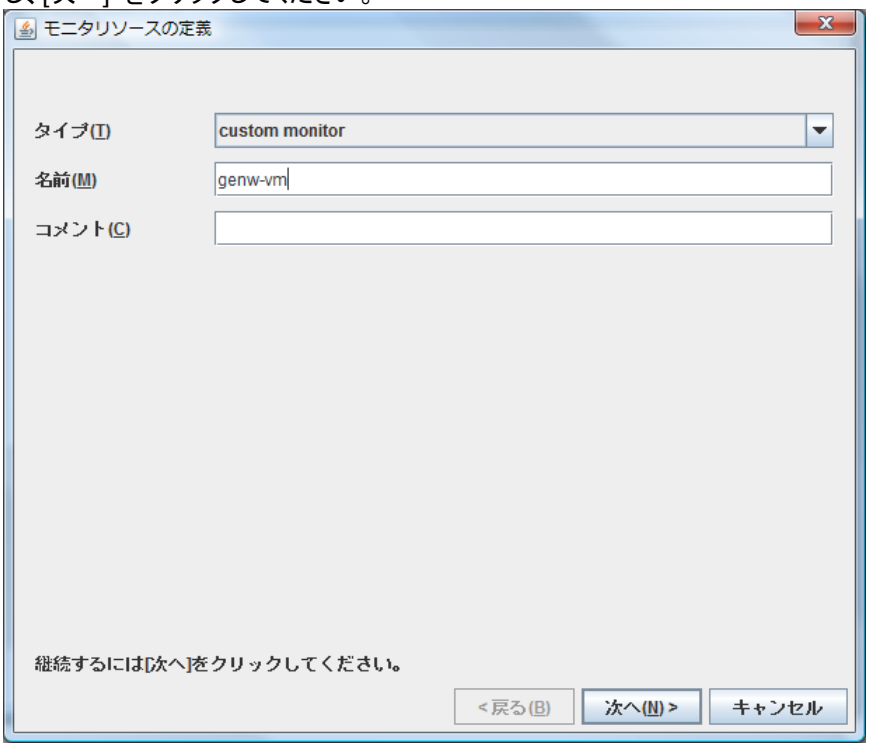
仮想マシン監視用モニタの設定

カスタムモニタリソースを利用する場合

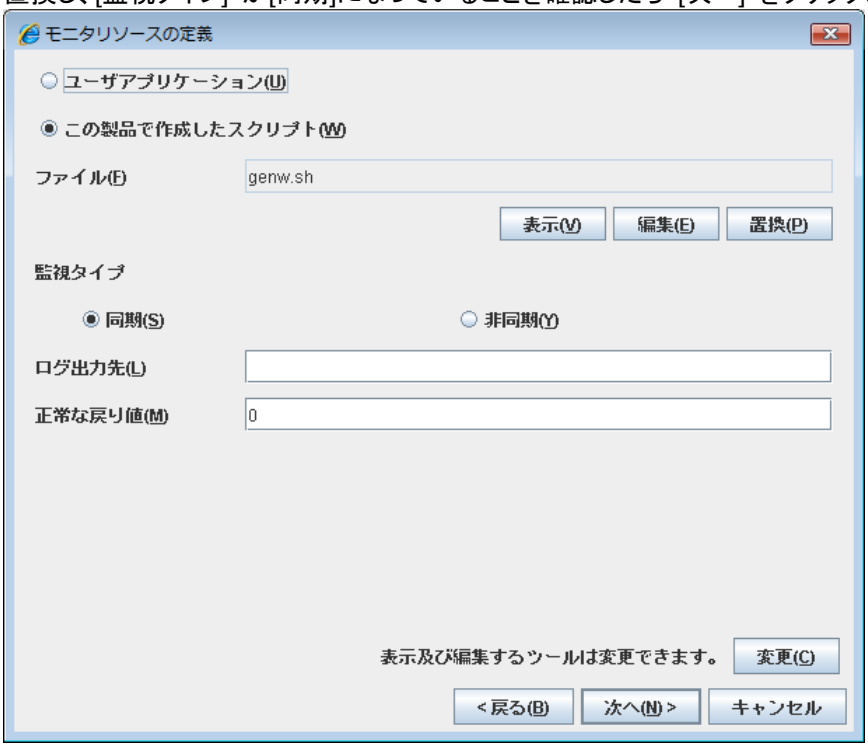
※ CLUSTERPRO X 3.0以降をご利用の場合、仮想マシンモニタリソースの利用を推奨します。

(1) CLUSTERPRO Builder の画面で、ツリービューの [Monitors] を右クリックし、[モニタリソースの追加] をクリックしてください。

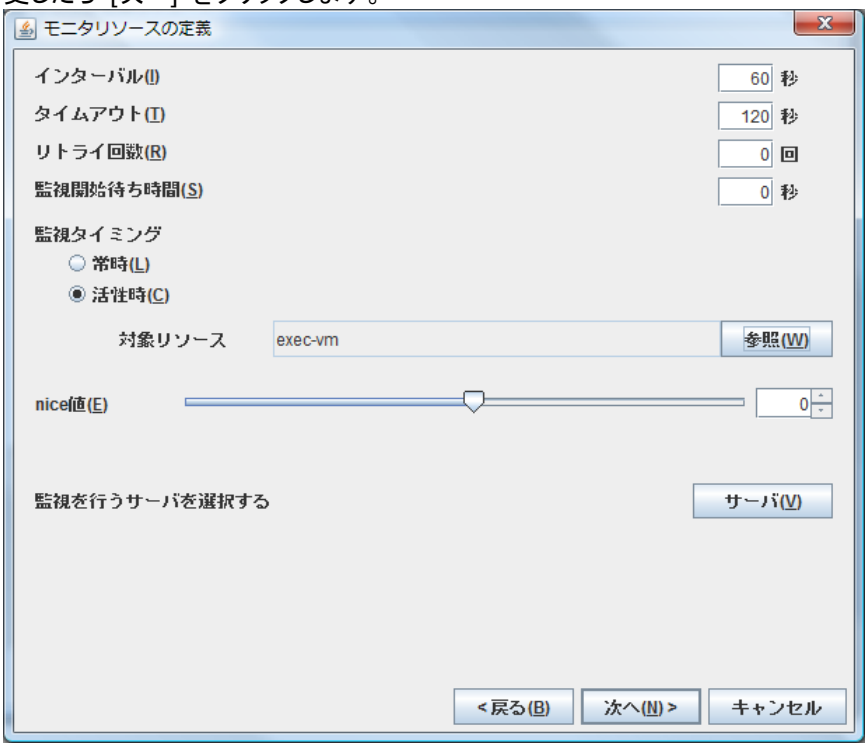
(2) 仮想マシン監視用カスタムモニタリソースを登録します。[タイプ] から [custom monitor resource] を選択してください。カスタムモニタリソースに任意の名前 (genw-vm) を設定し、[次へ] をクリックしてください。



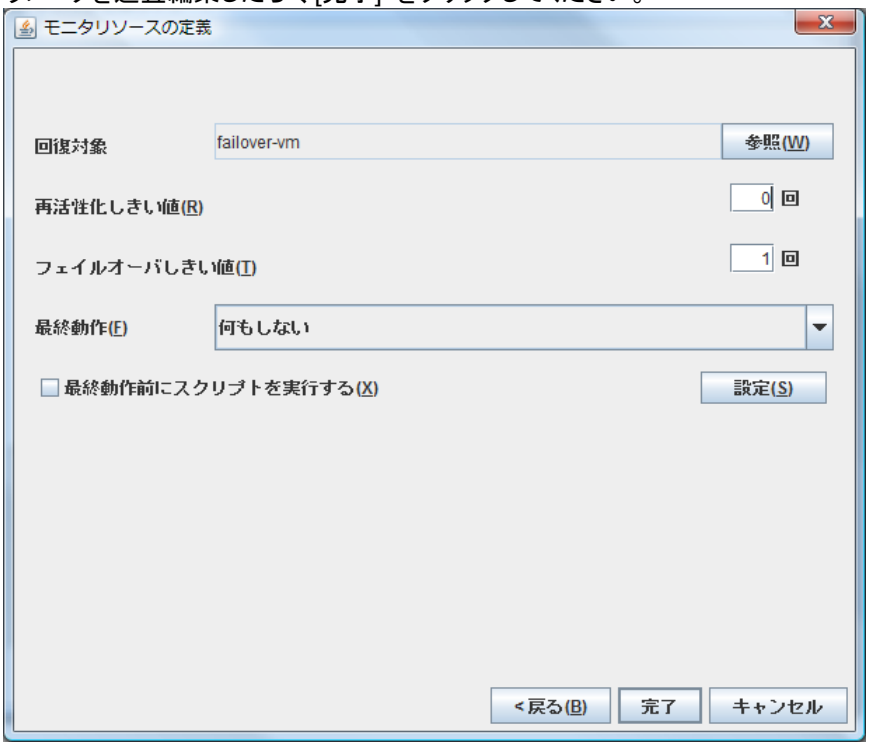
(3) [置換] をクリックし、genw.sh を本書のサンプルスクリプト clpvmmon.pl (78ページ) で置換し、[監視タイプ] が[同期]になっていることを確認したら [次へ] をクリックしてください。



(4) [監視タイミング] に [活性時] を選択し、対象リソースに仮想マシン制御用リソース [exec-vm] を選択し、[タイムアウト]、[インターバル]、[リトライ回数] を環境に合わせて変更したら [次へ] をクリックします。



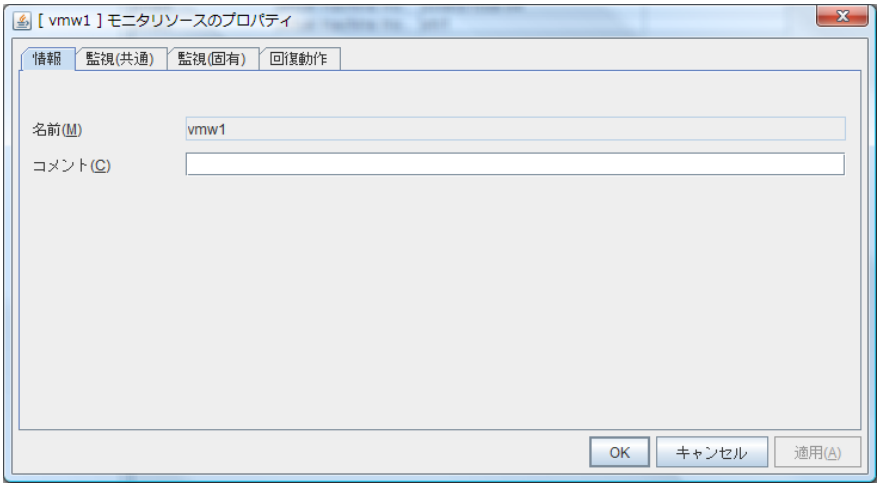
(5) [回復対象]として仮想マシン制御用リソース (exec-vm) が属するフェイルオーバグループ名 (failover-vm) を選択し、[OK] をクリックしてください。[再活性化しきい値] などのパラメータを適宜編集したら[1]、[完了] をクリックしてください。

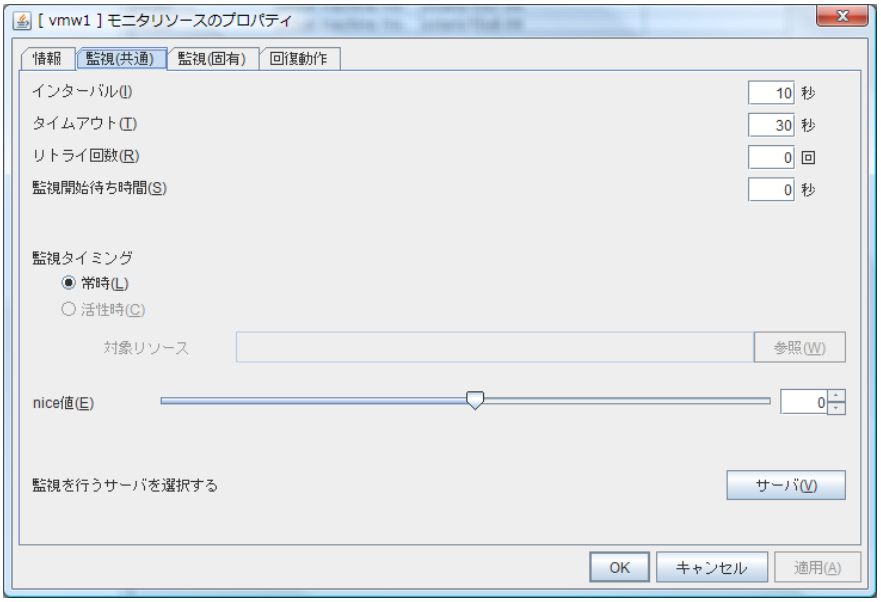


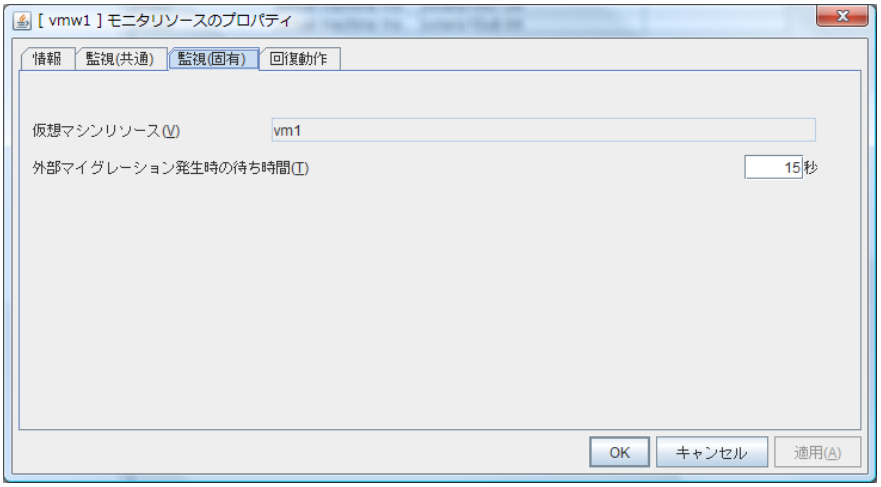
[1] 『CLUSTERPRO X リファレンスガイド 第 6 章 モニタリソースの詳細』を参考に、各パラメータを調整してください。

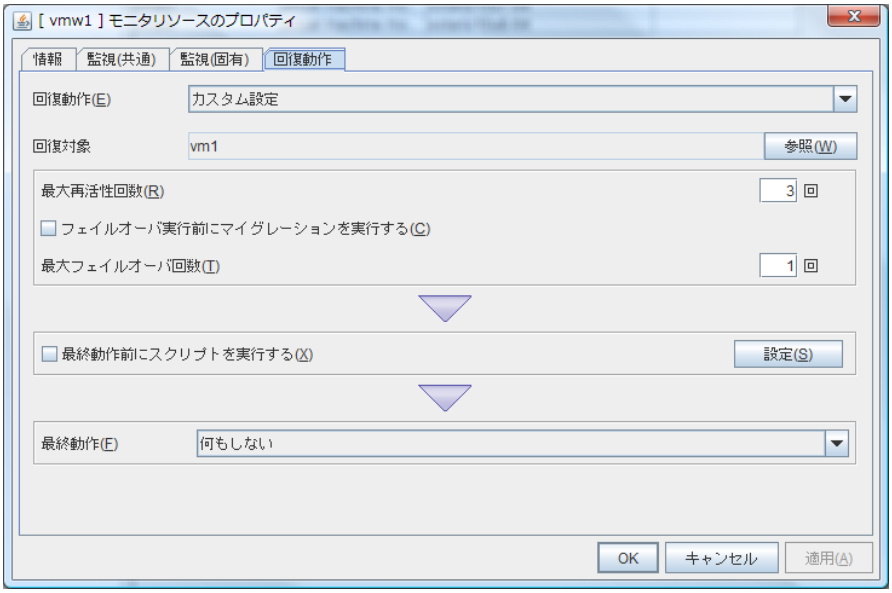
仮想マシンモニタリソースを利用する場合(X3.0以降)

(1) 仮想マシンリソースが作成されると自動的に仮想マシンモニタリソースが作成されます。
仮想マシンモニタリソースのプロパティで、変更が必要な項目を更新してください。








[vmw1] モニタリソースのプロパティ
情報
監視(共通)
監視(固有)
回復動作
回復動作(E)
カスタム設定
回復対象
vm1
参照(W)
最大再活性回数(R)
3
フェイルオーバ実行前にマイグレーションを実行する(C)
最大フェイルオーバ回数(T)
1
最終動作前にスクリプトを実行する(X)
設定(S)
最終動作(F)
何もしない
OK
キャンセル
適用(A)

VMware HAとの連携の設定

ホスト上のCLUSTERPRO X SingleServerSafeとVMware HAを連携させる

(1) 必要な数の仮想マシンを作成します。

(2) vCenter から VMware HA を構築します。

A) インベントリ上のデータデータセンタを右クリックして、[新規クラスタ]を選択し、クラスタを作成します。

B) インベントリ上のホストを作成したクラスタへドラッグ &ドロップし、ホストをクラスタへ参加させます。ゲスト OS のフェイルオーバ先にしたいホストを全てクラスタに参加させてください。

C) 作成したクラスタを右クリックして、[設定の編集]を選択し、クラスタの設定画面を表示します。

D) クラスタの設定画面上の左ペイン[クラスタ機能]を選択し、[VMware HA の有効化]をチェックします。

E) VMware HA による仮想マシンの監視が必要であれば、クラスタの設定画面上の左ペイン[VMware HA]->[仮想マシンの監視]で[仮想マシンの監視ステータス]欄の[仮想マシンの監視を有効にする]をチェックします。[1]

(3) 『CLUSTERPRO X インストール & 設定ガイド』に従い、VMware ESX の Service Console 上に CLUSTERPRO X SingleServerSafe (SSS) をインストールします。

(4) インストールした SSS に、ゲスト OS が格納される共有ディスクを監視するよう diskw を設定します。diskw の監視タイプには「READ(O_DIRECT)」、最終動作には「OS シャットダウン」を選択してください。また、ユーザ空間モニタリソース及びシャットダウンストール監視を利用する場合は、監視方法を「keepalive」に設定してください。

(5) 任意でその他の監視を設定します。

[1] 本設定を有効にするには、ゲスト上へVMware Toolsをインストールし動作させる必要があります。

ゲスト上のCLUSTERPRO X及びホスト上のCLUSTERPRO X SingleServerSafeとVMware HAを連携させる

(1) 『ホスト上の CLUSTERPRO X SingleServerSafe と VMware HA を連携させる (67ページ)』の(2)まで実行します。

(2) 『CLUSTERPRO X インストール & 設定ガイド』に従い、ゲスト OS に CLUSTERPRO をインストールしてください。

(3) 『CLUSTERPRO X インストール & 設定ガイド』に従い、CLUSTERPRO Builder でクラスタを構築してください。

(4) 『ホスト上の CLUSTERPRO X SingleServerSafe と VMware HA を連携させる (67ページ)』の(3)以降を実行します。

付録 A　　サンプルスクリプト

注:本ガイドのスクリプトをコピー＆ペーストした場合、全角スペースなどの意図しない文字が入る場合があります。

vmpower.start.pl

仮想マシンを起動するためのスクリプトです。**アンダーライン**の箇所を適宜編集してお使いください。

```
#!/usr/bin/perl -w

#
# Script for power on the Virtual Machine
#

use strict;

#--------------------------------------------------------------------------------
# Configuration
#--------------------------------------------------------------------------------
# The path to VM configuration file. This must be absolute UUID-based path.
my $cfg_path = "/vmfs/volumes/<datastore-uuid>/vm1/vm1.vmx";

# The interval to check the vm status. (second)
my $interval = 1;
# The miximum count to check the vm status.
my $max_cnt = 100;
# The timeout to start the vm. (second)
my $start_to = 10;
#--------------------------------------------------------------------------------
my $vmname = $cfg_path; # VMname to be outputted on log.
$vmname =~ s/^(.*¥/)(.*)(¥.vmx)/$2/;

# VM operation command path
my $vmcmd = "/usr/bin/vmware-cmd";

# VM execution state map
my %state = (
   "VM_EXECUTION_STATE_ON" => "on",
   "VM_EXECUTION_STATE_OFF" => "off",
   "VM_EXECUTION_STATE_SUSPENDED" => "suspended",
   "VM_EXECUTION_STATE_STUCK" => "stuck",
   "VM_EXECUTION_STATE_UNKNOWN" => "unknown"
);

#--------------------------------------------------------------------------------
# Main
#--------------------------------------------------------------------------------
exit 1 if (!&RegisterVm());

if (&IsPoweredOn()){
   exit 0;
}
```

```
else{
   if (&PowerOn()){
      if (&WaitPoweredOnDone()){
         exit 0;
      }
      else{
         exit 1;
      }
   }
   else{
      exit 1;
   }
}

#-------------------------------------------------------------------------------
# Functions
#-------------------------------------------------------------------------------
sub RegisterVm{
   my $svop = "-s register";
   my $vmcmd_list = $vmcmd . " -l";
   my @vmlist = `$vmcmd_list`;
   my $ret = 0;
   my $opn_ret;
   my $line;

   foreach (@vmlist){
      if (/$cfg_path/){
         &Log("[I] [$vmname] at localhost already registered.¥n");
         return 1;
      }
   }

   $opn_ret = open(my $fh, $vmcmd . " " . $svop . " " . $cfg_path . " 2>&1 |");
   if (!$opn_ret){
      &Log("[E] [$vmname] at localhost: $vmcmd $svop could not be executed.¥n");
      return 0;
   }

   $line = <$fh>;
   if (defined($line)){
      &Log("[E] [$vmname] at localhost: Could not register VM: $line¥n");
   }else{
      $ret = 1;
      &Log("[I] [$vmname] at localhost: Registered.¥n");
   }

   close($fh);

   return $ret;
}
#-------------------------------------------------------------------------------
sub IsPoweredOn{

   if (&IsEqualState($state{"VM_EXECUTION_STATE_ON"})){
      return 1;
   }else{
      return 0;
   }
```

```
}
#--------------------------------------------------------------------------------
sub IsEqualState{
   my $vmop = "getstate";
   my $state = shift;
   my $ret = 0;
   my $opn_ret;
   my $line;

   $opn_ret = open(my $fh, $vmcmd . " " . $cfg_path . " " . $vmop . " 2>&1 |");
   if (!$opn_ret){
      &Log("[E] [$vmname] at localhost: $vmcmd $vmop could not be executed.¥n");
      return 0;
   }

   $line = <$fh>;
   if (defined($line)){
      chomp($line);
      if ($line =~ /^$vmop¥(¥)¥s=¥s(.+)$/){
         $ret = 1 if ($1 eq $state);
         &Log("[D] [$vmname] at localhost: VM execution state is $1.¥n");
      }else{
         &Log("[E] [$vmname] at localhost: Could not get VM execution state: $line¥n");
      }
   }

   close($fh);

   return $ret;
}
#--------------------------------------------------------------------------------
sub PowerOn{
   my $vmop = "start";
   my $ret = 0;
   my $opn_ret;
   my $line;

   $opn_ret = open(my $fh, $vmcmd . " " . $cfg_path . " " . $vmop . " 2>&1 |");
   if (!$opn_ret){
      &Log("[E] [$vmname] at localhost: $vmcmd $vmop could not be executed.¥n");
      return 0;
   }

   eval{
      local $SIG{ALRM} = sub { die "timeout" };
      alarm($start_to);
      $line = <$fh>;
      alarm(0);
   };
   alarm(0);

   if ($@){
      if($@ =~ /timeout/){
         &Log("[E] [$vmname] at localhost: Cound not start VM: timeout($start_to
second)¥n");

         if (&IsEqualState($state{"VM_EXECUTION_STATE_STUCK"})){
            $ret = 1 if (&ResolveVmStuck());
```

```
        }
      }
   }
   else{
      if (defined($line)){
         chomp($line);
         if ($line =~ /^$vmop¥(¥)¥s=¥s(.+)$/){
            if ($1 == 1){
               $ret = 1;
               &Log("[I] [$vmname] at localhost: Started.¥n");
            }else{
               &Log("[E] [$vmname] at localhost: Cound not start VM: $1¥n");
            }
         }else{
            &Log("[E] [$vmname] at localhost: Cound not start VM: $line¥n");

            if (&IsEqualState($state{"VM_EXECUTION_STATE_STUCK"})){
               $ret = 1 if (&ResolveVmStuck());
            }
         }
      }

      close($fh);
   }

   return $ret;
}
#-------------------------------------------------------------------------------
sub ResolveVmStuck{
   my $vmop = "answer";
   my $ret = 0;
   my $opn_ret;
   my $line;

   $opn_ret = open(my $fh, "| ". $vmcmd . " " . $cfg_path . " " . $vmop);
   if (!$opn_ret){
      &Log("[E] [$vmname] at localhost: $vmcmd $vmop could not be executed.¥n");
      return 0;
   }

   # Answering "1) I _moved it" to keep vm config.
   print($fh "1¥n");

   close($fh);

   if (&IsEqualState($state{"VM_EXECUTION_STATE_STUCK"})){
      &Log("[E] [$vmname] at localhost: VM stuck could not be resolved.¥n");
   }else{
      $ret = 1;
      &Log("[I] [$vmname] at localhost: VM stuck is resolved.¥n");
   }

   return $ret;
}
#-------------------------------------------------------------------------------
sub WaitPoweredOnDone{

   for (my $i = 0; $i < $max_cnt; $i++){
```

```
        if (&IsEqualState($state{"VM_EXECUTION_STATE_ON"})){
            &Log("[I] [$vmname] at localhost: Powered on done. ($i)¥n");
            return 1;
        }
        sleep $interval;
    }

    &Log("[E] [$vmname] at localhost: Not powered on done. ($max_cnt)¥n");

    return 0;
}
#--------------------------------------------------------------------------------
sub Log{
    my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
    $year += 1900;
    $mon += 1;
    my $date = sprintf "%d/%02d/%02d %02d:%02d:%02d", $year, $mon, $mday, $hour, $min, $sec;
    print "$date $_[0]";
    return 0;
}
```

vmpower.stop.pl

仮想マシンを停止するためのスクリプトです。**アンダーライン**の箇所を適宜編集してお使いください。

```
#!/usr/bin/perl -w

#
# Script for power off the Virtual Machine
#

use strict;

#-------------------------------------------------------------------------------
# Configuration
#-------------------------------------------------------------------------------
# The path to VM configuration file. This must be absolute UUID-based path.
my $cfg_path = "/vmfs/volumes/<datastore-uuid>/vm1/vm1.vmx";

# The interval to check the vm status. (second)
my $interval = 5;
# The miximum count to check the vm status.
my $max_cnt = 100;
#-------------------------------------------------------------------------------
my $vmname = $cfg_path; # VMname to be outputted on log.
$vmname =~ s/^(.*¥/)(.*)(¥.vmx)/$2/;

# VM operation command path
my $vmcmd = "/usr/bin/vmware-cmd";

# VM execution state map
my %state = (
   "VM_EXECUTION_STATE_ON" => "on",
   "VM_EXECUTION_STATE_OFF" => "off",
   "VM_EXECUTION_STATE_SUSPENDED" => "suspended",
   "VM_EXECUTION_STATE_STUCK" => "stuck",
   "VM_EXECUTION_STATE_UNKNOWN" => "unknown"
);

#-------------------------------------------------------------------------------
# Main
#-------------------------------------------------------------------------------
if (&IsPoweredOn()){
   if (&PowerOff()){
      if (!&WaitPoweredOffDone()){
         exit 1;
      }
   }else{
      exit 1;
   }
}

if (&UnRegisterVm()){
   exit 0;
}else{
   exit 1;
```

```
}

#-------------------------------------------------------------------------------
# Functions
#-------------------------------------------------------------------------------
sub IsPoweredOn{

   if (&IsEqualState($state{"VM_EXECUTION_STATE_ON"})){
      return 1;
   }else{
      return 0;
   }
}
#-------------------------------------------------------------------------------
sub IsEqualState{
   my $vmop = "getstate";
   my $state = shift;
   my $ret = 0;
   my $opn_ret;
   my $line;

   $opn_ret = open(my $fh, $vmcmd . " " . $cfg_path . " " . $vmop . " 2>&1 |");
   if (!$opn_ret){
      &Log("[E] [$vmname] at localhost: $vmcmd $vmop could not be executed.¥n");
      return 0;
   }

   $line = <$fh>;
   if (defined($line)){
      chomp($line);
      if ($line =~ /^$vmop¥(¥)¥s=¥s(.+)$/){
         $ret = 1 if ($1 eq $state);
         &Log("[D] [$vmname] at localhost: VM execution state is $1.¥n");
      }else{
         &Log("[E] [$vmname] at localhost: Could not get VM execution state: $line¥n");
      }
   }

   close($fh);

   return $ret;
}
#-------------------------------------------------------------------------------
sub PowerOff{
   my $ret;

   # Soft stop.
   $ret = &PowerOffOpMode("soft");

   # Hard stop if Soft stop failed.
   if (!$ret){
      $ret = &PowerOffOpMode("hard");
   }

   return $ret;
}
#-------------------------------------------------------------------------------
sub PowerOffOpMode{
```

```
   my $vmop = "stop";
   my $powerop_mode = shift;
   my $ret = 0;
   my $opn_ret;
   my $line;

   return 0 if ($powerop_mode !~ /^hard|soft$/);

   $opn_ret = open(my $fh, $vmcmd . " " . $cfg_path . " " . $vmop . " " . $powerop_mode .
" 2>&1 |");
   if (!$opn_ret){
      &Log("[E] [$vmname] at localhost: $vmcmd $vmop $powerop_mode could not be
executed.¥n");
      return 0;
   }

   $line = <$fh>;
   if (defined($line)){
      chomp($line);
      if ($line =~ /^$vmop¥($powerop_mode¥)¥s=¥s(.+)$/){
         if ($1 == 1){
            $ret = 1;
            &Log("[I] [$vmname] at localhost: Stopped. ($powerop_mode)¥n");
         }else{
            &Log("[E] [$vmname] at localhost: Cound not stop ($powerop_mode) VM: $1¥n");
         }
      }else{
         &Log("[E] [$vmname] at localhost: Cound not stop ($powerop_mode) VM: $line¥n");
      }
   }else{
      if ($powerop_mode eq "soft"){
         $ret = 1;
         &Log("[I] [$vmname] at localhost: Stopped. ($powerop_mode)¥n");
      }
   }

   close($fh);

   return $ret;
}
#-------------------------------------------------------------------------------
sub WaitPoweredOffDone{

   for (my $i = 0; $i < $max_cnt; $i++){
      if (&IsEqualState($state{"VM_EXECUTION_STATE_OFF"})){
         &Log("[I] [$vmname] at localhost: Powered off done. ($i)¥n");
         return 1;
      }
      sleep $interval;
   }

   &Log("[E] [$vmname] at localhost: Not powered off done. ($max_cnt)¥n");

   return 0;
}
#-------------------------------------------------------------------------------
sub UnRegisterVm{
   my $svop = "-s unregister";
```

```
   my $vmcmd_list = $vmcmd . " -l";
   my @vmlist = `$vmcmd_list`;
   my $ret = 0;
   my $opn_ret;
   my $flag = 0;
   my $line;

   foreach (@vmlist){
      if (/$cfg_path/){
         $flag = 1;
      }
   }

   if ($flag == 0){
      &Log("[I] [$vmname] at localhost already unregistered.¥n");
      return 1;
   }else{
      $opn_ret = open(my $fh, $vmcmd . " " . $svop . " " . $cfg_path . " 2>&1 |");
      if (!$opn_ret){
         &Log("[E] [$vmname] at localhost: $vmcmd $svop could not be executed.¥n");
         return 0;
      }

      $line = <$fh>;
      if (defined($line)){
         &Log("[E] [$vmname] at localhost: Could not unregister VM: $line¥n");
      }else{
         $ret = 1;
         &Log("[I] [$vmname] at localhost: Unregistered.¥n");
      }

      close($fh);
   }

   return $ret;
}
#--------------------------------------------------------------------------------
sub Log{
   my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
   $year += 1900;
   $mon += 1;
   my $date = sprintf "%d/%02d/%02d %02d:%02d:%02d", $year, $mon, $mday, $hour, $min,
$sec;
   print "$date $_[0]";
   return 0;
}
```

clpvmmon.pl

仮想マシンの起動状態を確認するためのスクリプトです。**アンダーライン**の箇所を適宜編集してお使いください。

```
#!/usr/bin/perl -w

#
# Script for monitoring the Virtual Machine
#

use strict;

#-------------------------------------------------------------------------------
# Configuration
#-------------------------------------------------------------------------------
# The path to VM configuration file. This must be absolute UUID-based path.
my $cfg_path = "/vmfs/volumes/<datastore-uuid>/vm1/vm1.vmx";

# The interval to check the vm status. (second)
my $interval = 1;
#-------------------------------------------------------------------------------
my $vmname = $cfg_path; # VMname to be outputted on log.
$vmname =~ s/^(.*¥/)(.*)(¥.vmx)/$2/;

# VM operation command path
my $vmcmd = "/usr/bin/vmware-cmd";

# VM execution state map
my %state = (
   "VM_EXECUTION_STATE_ON" => "on",
   "VM_EXECUTION_STATE_OFF" => "off",
   "VM_EXECUTION_STATE_SUSPENDED" => "suspended",
   "VM_EXECUTION_STATE_STUCK" => "stuck",
   "VM_EXECUTION_STATE_UNKNOWN" => "unknown"
);

#-------------------------------------------------------------------------------
# Main
#-------------------------------------------------------------------------------
system("¥"ulimit¥" -s unlimited");
while (&IsHbIncrease()){
}
exit 0;

#-------------------------------------------------------------------------------
# Functions
#-------------------------------------------------------------------------------
sub IsHbIncrease{
   my $last_hb = -1;

   for (my $i = 0; $i < 2; $i++){
      if (&IsPoweredOn()){
         my $hb = &GetHb();
         if ($hb == -1){
            return 0;
```

```
            }

            if ($hb == $last_hb){
               &Log("[I] [$vmname] is stalled.¥n");
               return 0;
            }else{
               $last_hb = $hb;
            }
         }else{
            &Log("[I] [$vmname] is powered off.¥n");
            return 0;
         }
         sleep $interval;
      }

      return 1
}
#--------------------------------------------------------------------------------
sub GetHb{
      my $vmop = "getheartbeat";
      my $ret = -1;
      my $opn_ret;
      my $line;

      $opn_ret = open(my $fh, $vmcmd . " " . $cfg_path . " " . $vmop . " 2>&1 |");
      if (!$opn_ret){
         &Log("[E] [$vmname] at localhost: $vmcmd $vmop could not be executed.¥n");
         return -1;
      }

      $line = <$fh>;
      if (defined($line)){
         chomp($line);
         if ($line =~ /^$vmop¥(¥)¥s=¥s(.+)$/){
            $ret = $1;
#           &Log("[D] [$vmname] at localhost: Got VM heartbeat count $1.¥n");
         }else{
            &Log("[E] [$vmname] at localhost: Could not get VM heartbeat count: $line¥n");
         }
      }

      close($fh);

      return $ret;
}
#--------------------------------------------------------------------------------
sub IsPoweredOn{

      if (&IsEqualState($state{"VM_EXECUTION_STATE_ON"})){
         return 1;
      }else{
         return 0;
      }
}
#--------------------------------------------------------------------------------
sub IsEqualState{
      my $vmop = "getstate";
      my $state = shift;
```

```
   my $ret = 0;
   my $opn_ret;
   my $line;

   $opn_ret = open(my $fh, $vmcmd . " " . $cfg_path . " " . $vmop . " 2>&1 |");
   if (!$opn_ret){
      &Log("[E] [$vmname] at localhost: $vmcmd $vmop could not be executed.¥n");
      return 0;
   }

   $line = <$fh>;
   if (defined($line)){
      chomp($line);
      if ($line =~ /^$vmop¥(¥)¥s=¥s(.+)$/){
         $ret = 1 if ($1 eq $state);
#        &Log("[D] [$vmname] at localhost: VM execution state is $1.¥n");
      }else{
         &Log("[E] [$vmname] at localhost: Could not get VM execution state: $line¥n");
      }
   }

   close($fh);

   return $ret;
}
#--------------------------------------------------------------------------------
sub Log{
   my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
   $year += 1900;
   $mon += 1;
   my $date = sprintf "%d/%02d/%02d %02d:%02d:%02d", $year, $mon, $mday, $hour, $min,
$sec;
   print "$date $_[0]";
   return 0;
}
```

vmpreaction.sh

ゲスト OS 側の CLUSTERPRO からホスト OS 側の CLUSTERPRO にフェイルオーバ要求を発行する Linux 用スクリプトです。**アンダーライン**の箇所を適宜編集してお使いください。

1. ホスト OS の CLUSTERPRO のバージョンが X2.1 の場合

```
#! /bin/sh
#************************************************
#*                preaction.sh                  *
#************************************************
ulimit -s unlimited

echo START
echo $CLP_MONITORNAME

# 仮想マシン制御用グループリソース名
CLPRSC="exec_vm"
# 各ホストのIPアドレス(カンマ区切り)
CLPIP="10.0.0.1,10.0.0.2"

/opt/nec/clusterpro/bin/clptrnreq -t GRP_FAILOVER -r $CLPRSC -h $CLPIP
exit 0
```

2. ホスト OS の CLUSTERPRO のバージョンが X3.0 以降の場合

```
#! /bin/sh
#************************************************
#*                preaction.sh                  *
#************************************************
ulimit -s unlimited

echo START
echo $CLP_MONITORNAME

# 仮想マシン制御用グループリソース名
CLPRSC="vm1"
# 各ホストのIPアドレス(カンマ区切り)
CLPIP="10.0.0.1,10.0.0.2"

/opt/nec/clusterpro/bin/clprexec --failover -r $CLPRSC -h $CLPIP
exit 0
```

注:ゲスト OS の CLUSTERPRO のバージョンが X2.1 の場合、clprexec コマンドが存在しません。clptrnreq コマンドを使用するか、CLUSTERPRO CD から clprexec コマンドを取得して使用してください。

vmpreaction.bat

ゲストOS側のCLUSTERPROからホストOS側のCLUSTERPROにフェイルオーバ要求を発行するWindows 用バッチファイルです。**アンダーライン**の箇所を適宜編集してお使いください。

1. ホストOSのCLUSTERPROのバージョンがX2.1の場合

```
rem ****************************************
rem *            preaction.bat             *
rem ****************************************

echo START
echo %CLP_MONITORNAME%

rem 仮想マシン制御用グループリソース名
SET CLPRSC="exec-vm"
rem 各ホストのIPアドレス(カンマ区切り)
SET CLPIP="10.0.0.1,10.0.0.2"

clptrnreq.exe -t GRP_FAILOVER -r %CLPRSC% -h %CLPIP%
echo EXIT
```

2. ホストOSのCLUSTERPROのバージョンがX3.0以降の場合

```
rem ****************************************
rem *            preaction.bat             *
rem ****************************************

echo START
echo %CLP_MONITORNAME%

rem 仮想マシン制御用グループリソース名
SET CLPRSC="vm1"
rem 各ホストのIPアドレス(カンマ区切り)
SET CLPIP="10.0.0.1,10.0.0.2"

clprexec.exe --failover -r %CLPRSC% -h %CLPIP%
echo EXIT
```

注:ゲストOSのCLUSTERPROのバージョンがX2.1の場合、clprexecコマンドが存在しません。clptrnreqコマンドを使用するか、CLUSTERPRO CDからclprexecコマンドを取得して使用してください。

clphostmon_wrap.sh

clphostmon.plを使用するためのラッパースクリプトです。**アンダーライン**の箇所を適宜編集してお使いください。

```
#!/bin/sh

#
# Wrapper script for clphostmon.pl
#

#-------------------------------------------------------------------------------
# Configuration
#-------------------------------------------------------------------------------
# Specify the path of clphostmon.pl. It must be absolute path.
# e.g CLPHOSTMON_PATH="/opt/nec/clusterpro/work/clphostmon/clphostmon.pl"
CLPHOSTMON_PATH="absolute path of clphostmon.pl"
#-------------------------------------------------------------------------------

export PERL_LWP_SSL_VERIFY_HOSTNAME=0
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/vmware/vma/lib64:/opt/vmware/vma/lib
perl $CLPHOSTMON_PATH
```

clphostmon.pl

ホストの状態を確認するためのスクリプトです。**アンダーライン**の箇所を適宜編集してお使いください。

```
# hostname
hostname … ①
# vifp listservers
esxi hostname … ②   ESXi
```

```
#!/usr/bin/perl -w

#
# Script for monitoring the hardware
#


use strict;
use warnings;
use VMware::VILib;
use VMware::VIRuntime;
use VMware::VmaTargetLib;


#--------------------------------------------------------------------------------
# Configuration
#--------------------------------------------------------------------------------
# vMA_and_target_host : vMA hostname and ESXi Host name
# e.g my @vMA_and_target_host1 = ("vMA_host1", "ESXi_host1");
# monitoring_disks_on_host : The path to disk device file. Disk device path
#                            must be absolute path.
#                            We can set plural values in this variable.
#                            If we do not want to monitor disks, we should set
#                            this variable to undef as follows.
# e.g my @monitoring_disks_on_host1 = undef;
# monitoring_pnics_on_host : vmnic name. e.g vmnic0
#                            We can set plural values in this variable.
#                            If we do not want to monitor nics, we should set
#                            this variable to undef as follows.
# e.g my @monitoring_pnics_on_host1 = undef;

# Configuration for server1
my @vMA_and_target_host1 = ("hostname of vma … ①", "hostname of esxi … ②");
my @monitoring_disks_on_host1 = ("device file path", …);
my @monitoring_pnics_on_host1 = ("vmnic name",…);

# Configuration for server2
my @vMA_and_target_host2 = ("hostname of vma", "hostname of esxi");
my @monitoring_disks_on_host2 = ("device file path",…);
my @monitoring_pnics_on_host2 = ("vmnic name", …);

my @search_host = ([@vMA_and_target_host1], [@vMA_and_target_host2]);
my @search_disks = ([@monitoring_disks_on_host1], [@monitoring_disks_on_host2]);
my @search_pnics = ([@monitoring_pnics_on_host1],[@monitoring_pnics_on_host2]);
#--------------------------------------------------------------------------------

my $target_host;
```

```
my @monitoring_disks;
my @monitoring_pnics;

my $EXIT_CODE = 0;
my $TRACE = (1 << 0);
my $INFO = (1 << 1);
my $WARN = (1 << 2);
my $ERR = (1 << 3);
my $LOG_RELEASE = ($INFO | $WARN | $ERR);

#--------------------------------------------------------------------------------
# Main
#--------------------------------------------------------------------------------
chomp(my $local_host = `hostname`);
log_write($INFO, "Monitoring start on $local_host.¥n");

for (my $i= 0; $i < $#search_host + 1; $i++) {
   if($search_host[$i][0] eq $local_host) {
      $target_host = $search_host[$i][1];
      @monitoring_disks = @{$search_disks[$i]};
      @monitoring_pnics = @{$search_pnics[$i]};
      last;
   }
}
if (!defined($target_host)) {
   log_write($ERR,
      "$local_host was not found in specified vMA array. ¥n");
   exit (1);
}

my $host_exist;
my @esx_lists = VmaTargetLib::enumerate_targets();
foreach my $my_esx (@esx_lists) {
   if ($my_esx->name() eq $target_host) {
      $host_exist = 1;
      last;
   }
}
if (defined($host_exist)) {
   log_write($INFO, "Connecting target server : $target_host¥n");
} else {
   log_write($ERR, "Could not find target server in credential store.¥n");
   exit (1);
}
my $fast_login = VmaTargetLib::query_target($target_host);
$fast_login->login();
log_write($INFO, "Connected.¥n");

my $host = Vim::find_entity_view(view_type => 'HostSystem');
if (!defined($host)) {
   log_write($ERR, "Could not get HostSystem view.¥n");
   exit (1);
}

eval {
   if (defined($monitoring_disks[0])) {
      $EXIT_CODE = MonitoringDisk($host, @monitoring_disks);
   }
```

```
};
if ($@) {
   log_write($ERR, "Exception error occurred:¥n $@¥n");
   $EXIT_CODE = 1;
}

eval {
   if (defined($monitoring_pnics[0])) {
      $EXIT_CODE = MonitoringNetwork($host, @monitoring_pnics);
   }
};
if ($@) {
   log_write($ERR, "Exception error occurred:¥n $@¥n");
   $EXIT_CODE = 1;
}

$fast_login->logout();
log_write($INFO, "Monitoring completed:$EXIT_CODE¥n");
exit ($EXIT_CODE);

#-------------------------------------------------------------------------------
# Functions
#-------------------------------------------------------------------------------
sub MonitoringDisk {
   my ($hview, @target_disks) = @_;
   my $RESULT = 0;

   log_write($INFO, "Monitoring storage start.¥n");
   my $storage =
      Vim::get_view(mo_ref => $hview->configManager->storageSystem);

   log_write($INFO, "Refresh storage information.¥n");
   $storage->RefreshStorageSystem();
   log_write($INFO, "Rescan HBAs.¥n");
   $storage->RescanAllHba();

   foreach my $disk (@target_disks) {
      log_write($INFO, "Access target disk: $disk.¥n");
      my $part_info =
          $storage->RetrieveDiskPartitionInfo(devicePath => $disk);
      if ($#$part_info < 0) {
         log_write($ERR, "Could not access target disk.¥n");
         $RESULT = 1;
      }
      foreach (@$part_info) {
         log_write($INFO, "Access succeeded.¥n");
         my $disk_size =
            ConvirtUnit($_->spec->totalSectors * 512);
         log_write($INFO, "Disk Size is $disk_size ¥n");
      }
   }
   log_write($INFO, "Monitoring storage completed.¥n");
   return $RESULT;
}
#-------------------------------------------------------------------------------
sub MonitoringNetwork {
   my ($hview, @target_pnics) = @_;
   my $RESULT = 0;
```

```
   log_write($INFO, "Monitoring network start.¥n");
   my $network =
      Vim::get_view(mo_ref => $hview->configManager->networkSystem);

   log_write($INFO, "Refresh network information.¥n");
   $network->RefreshNetworkSystem();

   log_write($INFO, "Collect the nic status.¥n");
   my $pnics = $network->networkInfo->pnic;
   foreach my $target_pnic (@target_pnics) {
      my $exist = 0;
      foreach my $pnic (@$pnics) {
         if ($target_pnic eq $pnic->device) {
            $exist = 1;
            my $link = $pnic->linkSpeed;
            if (defined($link)) {
               log_write($INFO,
                   "$target_pnic : Link is Up.¥n");
            } else {
               log_write($ERR,
                   "$target_pnic : Link is Down.¥n");
               $RESULT = 1;
            }
         }
      }
      if ($exist == 0) {
         log_write($ERR, "$target_pnic : Missing device.¥n");
         $RESULT = 1;
      }
   }
   log_write($INFO, "Monitoring network completed.¥n");
   return $RESULT;
}
#-------------------------------------------------------------------------------
sub ConvirtUnit {
   my ($size) = @_;
   my $convirt_size;

   if ($size <= 0 || $size eq '') {
      return $size;
   }

   if ($size < 1024) {
      $convirt_size = $size . "Bytes";
   } elsif ($size < 1024 * 1024) {
      $convirt_size = sprintf("%.2f", $size / 1024) . "KBytes";
   } elsif ($size < 1024 * 1024 * 1024) {
      $convirt_size = sprintf("%.2f",
         $size / (1024 * 1024)) . "MBytes";
   } elsif ($size < 1024 * 1024 * 1024 * 1024) {
      $convirt_size = sprintf("%.2f",
         $size / (1024 * 1024 * 1024)) . "GBytes";
   } else {
      $convirt_size = sprintf("%.2f",
         $size / (1024 * 1024 * 1024)) . "TBytes";
   }
```

```
    return $convirt_size;
}
#--------------------------------------------------------------------------------
sub log_write {
    my ($level, $string) = @_;
    my $pre_level;

    if ($level & $LOG_RELEASE) {
        if ($level == $TRACE) {
            $pre_level = "T";
        } elsif ($level == $INFO) {
            $pre_level = "I";
        } elsif ($level == $WARN) {
            $pre_level = "W";
        } elsif ($level == $ERR) {
            $pre_level = "E";
        } else {
            $pre_level = "U";
        }
    } else {
        # not print log
        return 0;
    }

    my $ctime = get_ctime();
    my $head = sprintf("%s %s : ", $pre_level, $ctime);

    print "$head $string";

    return 0;
}
#--------------------------------------------------------------------------------
sub get_ctime {
    my $time = time;
    my ($sec, $min, $hour, $day, $month, $year) = localtime($time);
    $year += 1900;
    $month += 1;
    my $ctime = sprintf ("%d/%02d/%02d %02d:%02d:%02d",
          $year, $month, $day, $hour, $min, $sec);
    return $ctime;
}
#--------------------------------------------------------------------------------
```

stdnhost_wrap.sh

shutdownhost.plを使用するためのラッパースクリプトです。**アンダーライン**の箇所を適宜編集してお使いください。

```
#!/bin/sh

#
# Wrapper script for shutdownhost.pl
#

#-----------------------------------------------------------------------------
# Configuration
#-----------------------------------------------------------------------------
# Specify the path of shutdownhost.pl. It must be absolute path.
# e.g STDNHOST_PATH="/opt/nec/clusterpro/work/shutdownhost/shutdownhost.pl"
STDNHOST_PATH="absolute path of shutdownhost.pl"
#-----------------------------------------------------------------------------

export PERL_LWP_SSL_VERIFY_HOSTNAME=0
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/vmware/vma/lib64:/opt/vmware/vma/lib
perl $STDNHOST_PATH
```

shutdownhost.pl

ホストをシャットダウンするためのスクリプトです。**アンダーライン**の箇所を適宜編集してお使いください。

 # hostname
[hostname] … ①

 # vifp listservers
[esxi hostname] … ②　ESXi

```
#!/usr/bin/perl -w

#
# Script for shutting down ESXi host;
#


use strict;
use warnings;
use VMware::VILib;
use VMware::VIRuntime;
use VMware::VmaTargetLib;

#--------------------------------------------------------------------------------
# Configuration
#--------------------------------------------------------------------------------
# vMA_and_target_host : vMA hostname and ESXi Host name
# e.g my @vMA_and_target_host1 = ("vMA_host1", "ESXi_host1");

# Configuration for server1
my @vMA_and_target_host1 = ("hostname of vma … ①", "hostname of esxi … ②");

# Configuration for server2
my @vMA_and_target_host2 = ("hostname of vma", "hostname of esxi");

my @search_host = ([@vMA_and_target_host1], [@vMA_and_target_host2]);

# Specify log path.
# e.g my $log_path = "/opt/nec/clusterpro/log/shutdownhost.log";
my $log_path = "log file path";
#--------------------------------------------------------------------------------

my $target_host;
my $EXIT_CODE = 0;

my $TRACE = (1 << 0);
my $INFO = (1 << 1);
my $WARN = (1 << 2);
my $ERR = (1 << 3);
my $LOG_RELEASE = ($INFO | $WARN | $ERR);

#--------------------------------------------------------------------------------
# Main
#--------------------------------------------------------------------------------

log_write($INFO, "Shutdown the host starting.¥n");
```

```
eval {
   chomp(my $local_host = `hostname`);
   for (my $i= 0; $i < $#search_host + 1; $i++) {
      if($search_host[$i][0] eq $local_host) {
         $target_host = $search_host[$i][1];
         last;
      }
   }
   if (!defined($target_host)) {
      log_write($ERR,
         "$local_host was not found in specified vMA array. ¥n");
      exit (1);
   }

   my $host_exist;
   my @esx_lists = VmaTargetLib::enumerate_targets();
   foreach my $my_esx (@esx_lists) {
      if ($my_esx->name() eq $target_host) {
         $host_exist = 1;
         last;
      }
   }
   if (defined($host_exist)) {
      log_write($INFO, "Connecting target server : $target_host¥n");
   } else {
      log_write($ERR,
         "Could not find target server in credential store.¥n");
      exit (1);
   }

   my $fast_login = VmaTargetLib::query_target($target_host);
   $fast_login->login();
   log_write($INFO, "Connected.¥n");

   log_write($INFO,
      "Sending request to shutdown $target_host from $local_host¥n");
   my $host = Vim::find_entity_view(view_type => 'HostSystem');
   $host->ShutdownHost(force => 1);

   log_write($INFO, "Done.¥n");
   $fast_login->logout();
};
if ($@) {
   log_write($ERR, "Exception error occurred:¥n $@¥n");
   $EXIT_CODE = 1;
}

exit ($EXIT_CODE);
#--------------------------------------------------------------------------------
sub log_write {
   my ($level, $string) = @_;
   my $pre_level;

   if ($level & $LOG_RELEASE) {
      if ($level == $TRACE) {
         $pre_level = "T";
      } elsif ($level == $INFO) {
         $pre_level = "I";
```

```
      } elsif ($level == $WARN) {
         $pre_level = "W";
      } elsif ($level == $ERR) {
         $pre_level = "E";
      } else {
         $pre_level = "U";
      }
   } else {
      # not print log
      return 0;
   }

   my $ctime = get_ctime();
   my $head = sprintf("%s %s : ", $pre_level, $ctime);

   open(OUT, "+>>$log_path");
   print OUT "$head $string";
   close(OUT);

   return 0;
}
#--------------------------------------------------------------------------------
sub get_ctime {
   my $time = time;
   my ($sec, $min, $hour, $day, $month, $year) = localtime($time);
   $year += 1900;
   $month += 1;
   my $ctime = sprintf ("%d/%02d/%02d %02d:%02d:%02d",
         $year, $month, $day, $hour, $min, $sec);
   return $ctime;
}
#--------------------------------------------------------------------------------
```