

Linux 版 Horizon 7 バージョ ン 7.0.3 デスクトップのセッ トアップ

VMware Horizon 7 7.0.3

VMware Horizon 7 7.0.2

VMware Horizon 7 7.0.1



vmware®

最新の技術ドキュメントは、VMware の Web サイト (<https://docs.vmware.com/jp/>) でご確認ください。このドキュメントに関するご意見およびご感想は、docfeedback@vmware.com までお送りください。

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

VMware株式会社
105-0013 東京都港区浜松町 1-30-5
浜松町スクエア 13F
www.vmware.com/jp

Copyright © 2016 年 VMware, Inc. All rights reserved. [著作権および商標情報](#)。

目次

Horizon 7 バージョン 7.0.3 for Linux デスクトップのセットアップ 5

1 機能とシステムの要件 6

- Horizon Linux デスクトップの機能 6
- Horizon 7 for Linux デスクトップの構成手順の概要 9
- Horizon 7 for Linux のシステム要件 10
 - 2D または vSGA グラフィックスの仮想マシン設定 12

2 デスクトップ デプロイのための Linux 仮想マシンの準備 15

- 仮想マシンを作成して、Linux インストールする 15
- リモート デスクトップ デプロイ用の Linux マシンの準備 16
- Horizon Agent 用依存パッケージのインストール 18

3 Linux デスクトップの Active Directory 統合のセットアップ 21

- Linux と Active Directory の統合 21
- シングル サインオンとスマート カード リダイレクトのセットアップ 22

4 Linux デスクトップのグラフィックスのセットアップ 25

- vGPU を使用するための RHEL 6.6/6.7/6.8 と RHEL 7.2 の構成 25
 - NVIDIA GRID vGPU グラフィック カードの VIB の ESXi ホストへのインストール 26
 - Linux 仮想マシンで vGPU を使用するための共有 PCI デバイスの構成 27
 - NVIDIA GRID vGPU ディスプレイ ドライバのインストール 28
 - NVIDIA ディスプレイ ドライバがインストールされているかどうかの確認 30
- vDGA を使用するための RHEL 6.6/6.7/6.8 の構成 31
 - ホストで NVIDIA GRID を使用するために DirectPath I/O を有効にする 31
 - vDGA パススルー デバイスの RHEL 6.6/6.7/6.8 仮想マシンへの追加 32
 - vDGA 用の NVIDIA ディスプレイ ドライバのインストール 32
 - NVIDIA ディスプレイ ドライバがインストールされているかどうかの確認 34
- vSGA を使用するための RHEL 7.2 の構成 35
 - vSGA 用の NVIDIA グラフィック カードの VIB の ESXi ホストへのインストール 35
 - Linux 仮想マシンでの vSGA の 3D 機能の構成 36
 - vSGA が Linux 仮想マシンで実行されているかどうかの確認 37

5 Horizon Agent のインストール 39

- Linux 仮想マシンへの Horizon Agent のインストール 39
 - install_viewagent.sh コマンドライン オプション 40
- Linux Agent 用証明書の構成 42
- Linux 仮想マシンでの Horizon Agent のアップグレード 43

Linux 仮想マシンでの Horizon Agent のアップグレード	44
Horizon 7 for Linux マシンをアンインストール	45
6 Linux デスクトップの構成オプション	46
Linux デスクトップでの構成ファイルのオプション設定	46
Linux デスクトップの Blast 設定の例	50
Linux デスクトップの vSphere コンソールへの表示を抑制する	51
7 Linux デスクトップ プールの作成と管理	52
Linux 版手動デスクトップ プールの作成	52
Linux デスクトップ プールの管理	53
Linux の自動化された完全なクローン デスクトップ プールの作成	55
ブローカ PowerCLI コマンド	57
8 手動デスクトップ プールのための Horizon 7 の一括デプロイ	60
Linux デスクトップの一括デプロイの概要	60
Linux デスクトップの一括アップグレードの概要	62
Linux デスクトップ マシンのクローンを作成するために仮想マシン テンプレートを作成する	63
Linux デスクトップを展開するサンプル PowerCLI スクリプトの入力ファイル	65
Linux 仮想マシンのクローンを作成するサンプル スクリプト	65
クローン作成した仮想マシンを Active Directory ドメインに参加させるサンプル スクリプト	69
SSH を使用してクローン作成した仮想マシンを Active Directory ドメインに参加させるサンプル スクリプト	72
Linux 仮想マシンに構成ファイルをアップロードするサンプル スクリプト	76
SSH を使用して Linux 仮想マシンに構成ファイルをアップロードするサンプル スクリプト	79
Linux デスクトップ マシンで Horizon Agent をアップグレードするサンプル スクリプト	84
SSH を使用して Linux 仮想マシンで Horizon Agent をアップグレードするサンプル スクリプト	88
Linux 仮想マシンで操作を実行するサンプル スクリプト	94
9 Linux デスクトップのトラブルシューティング	99
Horizon 7 for Linux マシンの診断情報の収集	99
リモート デスクトップとクライアント ホストの間でのコピーと貼り付けに関するトラブルシューティング	100
TCP 接続の受信を許可するための Linux ファイアウォールの構成	100
View Agent が iPad Pro Horizon Client で切断できない	100
ドラッグアンドドロップ後、SLES 12 SP1 デスクトップが自動更新しない	101
SSO がパワーオフ エージェントに接続できない	101
Linux 版手動デスクトップ プール作成後の接続不能な仮想マシン	101

Horizon 7 バージョン 7.0.3 for Linux デスクトップのセットアップ

Horizon 7 for Linux デスクトップのセットアップドキュメントは、Linux ゲスト OS の準備、仮想マシンへの Horizon Agent のインストール、Horizon 7 の展開環境で使用するための View Administrator でのマシンの構成など、Linux 仮想マシンをセットアップして VMware Horizon 7[®] デスクトップとして使用するための情報について説明します。

対象読者

この情報は、Linux ゲスト OS で実行するリモート デスクトップを構成および使用するすべてのユーザーを対象にしています。これらの情報は、仮想マシン テクノロジーおよびデータセンターの運用に精通している経験豊富な Linux システム管理者向けに記述されています。

機能とシステムの要件

Horizon 6 以降では、ユーザーは Linux オペレーティング システムを実行しているリモート デスクトップに接続できます。

この章には、次のトピックが含まれています。

- [Horizon Linux デスクトップの機能](#)
- [Horizon 7 for Linux デスクトップの構成手順の概要](#)
- [Horizon 7 for Linux のシステム要件](#)

Horizon Linux デスクトップの機能

Horizon 7 では、Linux デスクトップの多くの新機能が導入されます。

Horizon 7.0.3 のリリースでは以下の新機能がサポートされています。

- オーディオ入力のサポート
- Ubuntu 16.04 のサポート
- 複数のモニタをサポートするソフトウェア H.264 エンコーダ
- すべてのディストリビューションでのクリップボード リダイレクトのサポート
- RHEL 6.6/6.7/6.8/7.2 Workstation x64 での NVIDIA M6 グラフィック カードによる vGPU のサポート

次のリストでは、Horizon Linux デスクトップの主な機能を示します。

自動化される完全なクローン デスクトップ プール	Horizon 7 バージョン 7.0.2 から、Linux デスクトップの自動化される完全なクローン デスクトップ プールを作成できます。
手動デスクトップ プール	マシン ソース <ul style="list-style-type: none">■ 管理対象仮想マシン - vCenter Server 仮想マシンのマシン ソース。管理対象仮想マシンは、新規およびアップグレードのデプロイにサポートされます。

- 管理対象外の仮想マシン - 他のソースのマシン ソース。管理対象外の仮想マシンのデプロイからアップグレードする場合のみ、管理対象外の仮想マシンはサポートされます。

注: 管理対象仮想マシンには、View 接続サーバ 7.0.1 またはそれ以降が必要です。

複数のモニタ

- vDGA/vGPU デスクトップは、最大 2560x1600 の解像度を 4 台のモニタでサポートします。
- vSphere 6.0 以降の 2D/vSGA デスクトップは、最大 2048x1536 の解像度を 4 台のモニタでサポートし、最大 2560x1600 の解像度を 3 台のモニタでサポートします。
- vSphere 5.5 U3 での 2D デスクトップ
 - RHEL/CentOS 6.6/6.7/6.8 および SLED 11 SP3/SP4 は、最大 2560x1600 の解像度を 4 台のモニタでサポートします。
 - Ubuntu 12.04/14.04/16.04、RHEL/CentOS 7.2、および SLES 12 SP1 は、最大 2048x1536 の解像度を 4 台のモニタ、最大 2560x1600 の解像度を 3 台のモニタでサポートします。

Ubuntu 14.04 および 16.04 で複数のモニタをサポートするには、Compiz を無効にする必要があります。詳細については、<http://kb.vmware.com/kb/2114809> を参照してください。

SLES 12 SP1 では、カーネル レベル kernel-default-3.12.49-11.1 のデフォルト パッケージを使用する必要があります。パッケージをアップグレードしている場合、マルチモニタ機能は動作せず、デスクトップは 1 台のモニタに表示されます。

可逆圧縮 PNG

デスクトップで生成される画像とビデオは、クライアント デバイスで正確なピクセル レベルで表示されます。

ソフトウェア H.264 エンコーダ

H.264 は、特に低い帯域幅ネットワークでは、Horizon デスクトップの Blast Extreme のパフォーマンスを改善できます。クライアント側が H.264 を無効にすると、Blast Extreme は自動的に JPEG/PNG のエンコーディングに戻ります。

Horizon 7 バージョン 7.0.3 では、複数のモニタがサポートされます。

3D グラフィックス

3D グラフィックスは、次の Linux バージョンとグラフィック カードの組み合わせでサポートされます。

- vSGA は、RHEL 7.2 Workstation x64 と NVIDIA GRID K1 または K2 のグラフィック カードの組み合わせでサポートされます。
- vDGA は、RHEL 6.6/6.7/6.8 Workstation x64 と NVIDIA GRID K1 または K2 のグラフィック カードの組み合わせでサポートされます。
- vGPU は、RHEL 6.6/6.7/6.8 Workstation x64 と NVIDIA Maxwell M60 グラフィック カードの組み合わせでサポートされます。

- vGPU は、RHEL 7.2 Workstation x64 と NVIDIA Maxwell M60 グラフィックカードの組み合わせでサポートされます。
- vGPU は、RHEL 6.6/6.7/6.8 Workstation x64 と NVIDIA M6 グラフィックカードの組み合わせでサポートされます。
- vGPU は、RHEL 7.2 Workstation x64 と NVIDIA M6 グラフィックカードの組み合わせでサポートされます。

クリップボード リダイレクト

クリップボード リダイレクトを使用すると、リッチ テキストまたはプレーン テキストをクライアント ホストとリモートの Linux デスクトップ間でコピーおよび貼り付けできます。Horizon Agent のオプションを使用して、コピー/貼り付けの方向と最大テキスト サイズを設定できます。

シングル サインオン

シングル サインオンは、次の Linux バージョンでサポートされます。

- RHEL 6.6/6.7/6.8 Workstation x64
- CentOS 6.6/6.7/6.8 x64
- SLED 11 SP3/SP4 x64

SSO でスマート カード リダイレクト

スマート カード リダイレクトは RHEL 6.6/6.7/6.8 Workstation x64 でサポートされます。Personal Identity Verification (PIV) カードと Common Access Card (CAC) がサポートされます。Mac クライアントはサポートされません。

オーディオ入力

クライアント ホストからリモート Linux デスクトップへのオーディオ入力リダイレクトがサポートされます。この機能は、USB リダイレクト機能をベースにしています。この機能を有効にするには、インストール中にこの機能を選択する必要があります。また、デバイスでシステムのデフォルト オーディオを設定し、アプリケーションでオーディオ入力に「PulseAudio サーバ (ローカル)」を選択する必要があります。この機能は次のディストリビューションでサポートされます。

- Ubuntu 14.04 x64
- Ubuntu 16.04 x64
- CentOS 7.2 x64
- RHEL 7.2 workstation x64

Linux デスクトップとデスクトップ プールには次の制限があります。

- USB リダイレクト、仮想印刷、ロケーションベースの印刷、リアルタイム ビデオはサポートされません。

注: セキュリティ サーバが利用される場合、社内のファイアウォールでポート 22443 を開き、セキュリティ サーバと Linux デスクトップ間のトラフィックを許可する必要があります。

Horizon 7 for Linux デスクトップの構成手順の概要

Horizon 7 for Linux デスクトップをインストールして構成する場合、仮想マシンに 2D グラフィックスまたは 3D グラフィックスをインストールするかどうかによって、実行する必要がある一連の手順が異なります。

2D グラフィックス - 構成手順の概要

2D グラフィックスの場合、次の手順を実行します。

- 1 Horizon 7 for Linux のデプロイ環境をセットアップするためのシステム要件を確認します。 [Horizon 7 for Linux のシステム要件](#)を参照してください。
- 2 vSphere で仮想マシンを作成し、Linux オペレーティング システムをインストールします。 [仮想マシンを作成して、Linux インストールする](#)を参照してください。
- 3 Horizon 7 環境でデスクトップとしてデプロイするゲスト OS を準備します。 [リモート デスクトップ デプロイ用の Linux マシンの準備](#)を参照してください。
- 4 Active Directory で認証するように Linux ゲスト OS を構成します。この手順は、ユーザー環境の要件を基準に、サードパーティ ソフトウェアを使用して実施します。この手順についてはこのガイドでは説明しません。
- 5 Linux 仮想マシンに Horizon Agent をインストールします。 [Linux 仮想マシンへの Horizon Agent のインストール](#)を参照してください。
- 6 構成した Linux 仮想マシンを含むデスクトップ プールを作成します。 [Linux 版手動デスクトップ プールの作成](#)を参照してください。

3D グラフィックス - 構成手順の概要

マシンに Horizon Agent をインストールし、View Administrator にデスクトップ プールをデプロイする前に、NVIDIA GRID vGPU、vDGA、または vSGA の構成を完了します。

- 1 Horizon 7 for Linux のデプロイ環境をセットアップするためのシステム要件を確認します。 [Horizon 7 for Linux のシステム要件](#)を参照してください。
- 2 vSphere で仮想マシンを作成し、Linux オペレーティング システムをインストールします。 [仮想マシンを作成して、Linux インストールする](#)を参照してください。
- 3 Horizon 7 環境でデスクトップとしてデプロイするゲスト OS を準備します。 [リモート デスクトップ デプロイ用の Linux マシンの準備](#)を参照してください。
- 4 Active Directory で認証するように Linux ゲスト OS を構成します。この手順は、ユーザー環境の要件を基準に、サードパーティ ソフトウェアを使用して実施します。この手順についてはこのガイドでは説明しません。
- 5 ESXi ホストと Linux 仮想マシンで 3D 機能を構成します。インストールする 3D 機能に関する手順を実行します。
 - [vGPU を使用するための RHEL 6.6/6.7/6.8 と RHEL 7.2 の構成](#)を参照してください。
 - [vDGA を使用するための RHEL 6.6/6.7/6.8 の構成](#)を参照してください。
 - [vSGA を使用するための RHEL 7.2 の構成](#)を参照してください。

- 6 Linux 仮想マシンに Horizon Agent をインストールします。[Linux 仮想マシンへの Horizon Agent のインストール](#)を参照してください。
- 7 構成した Linux 仮想マシンを含むデスクトップ プールを作成します。[Linux 版手動デスクトップ プールの作成](#)を参照してください。

一括デプロイ

View Administrator では、手動デスクトップ プールへの Linux 仮想マシンのデプロイのみを行うことができます。vSphere PowerCLI を使用すると、Linux デスクトップ マシンのプールのデプロイを自動化するスクリプトを開発できます。[8 章 手動デスクトップ プールのための Horizon 7 の一括デプロイ](#)を参照してください。

Horizon 7 for Linux のシステム要件

Horizon 7 for Linux は、特定のオペレーティング システム、Horizon 7、および vSphere プラットフォームの要件に適合している必要があります。

Horizon Agent でサポートされる Linux バージョン

次の表は、デスクトップ プール内の仮想マシンでサポートされる Linux オペレーティング システムを一覧で示しています。

表 1-1. View Agent でサポートされる Linux オペレーティング システム

Linux ディストリビューション	アーキテクチャ
Ubuntu 14.04 および 16.04	x64
注: パフォーマンスの低下を防ぐには、Compiz を無効にします。詳細については、 http://kb.vmware.com/kb/2114809 を参照してください。	
Ubuntu 12.04	x64
RHEL 6.6/6.7/6.8 および 7.2	x64
CentOS 6.6/6.7/6.8 および 7.2	x64
NeoKylin 6 Update 1	x64
SLED 11 SP3/SP4	x64
SLES 12 SP1	x64

注: Linux エージェントは、一部の Linux ディストリビューションで依存パッケージを使用します。詳細については、[Horizon Agent 用依存パッケージのインストール](#)を参照してください。

必須のプラットフォームと Horizon 7 ソフトウェア バージョン

Horizon 7 for Linux をインストールして使用するには、デプロイする環境が特定の vSphere プラットフォーム、Horizon 7、およびクライアント要件を満たしている必要があります。

表 1-2. 必須のプラットフォームと Horizon 7 ソフトウェア バージョン

プラットフォームとソフトウェア	サポートされているバージョン
vSphere プラットフォームのバージョン	vSphere 5.5 U3、vSphere 6.0 U2 以降のリリース、vSphere 6.5 以降のリリース。
Horizon 環境	■ Horizon 接続サーバ 7.0.3
Horizon Client ソフトウェア	<ul style="list-style-type: none"> ■ Horizon Client 4.3.0 for Android ■ Horizon Client 4.3.0 for Windows ■ Horizon Client 4.3.0 for Linux ■ Horizon Client 4.3.0 for Mac OS X ■ Horizon Client 4.3.0 for iOS (iPad Pro) ■ Chrome での HTML Access 4.3.0 ■ ゼロ クライアントはサポートされません

注: HTML Access はオーディオ出力をサポートしません。

Linux 仮想マシンにより使用される TCP ポート

View Agent と Horizon Client は、互いのネットワーク アクセスや各種 View server コンポーネント間のネットワーク アクセスに TCP ポートを使用します。

表 1-3. Linux 仮想マシンにより使用される TCP ポート

送信元	ポート	送信先	ポート	プロトコル	説明
Horizon Client	*	Linux Agent	22443	TCP	Blast Security Gateway が使用されない場合は Blast
セキュリティ サーバ、View 接続サーバ、または Access Point アプリアンス	*	Linux Agent	22443	TCP	Blast Security Gateway が使用される場合は Blast
View Agent	*	View 接続サーバ	4001、4002	TCP	JMS SSL トラフィック。

注: クライアントにより使用される TCP および UDP ポートの詳細については、View ドキュメントの『Horizon Client および View Agent のセキュリティ』で「クライアントおよび View Agent により使用される TCP および UDP ポート」を参照してください。

Linux 仮想マシンにより使用される Linux アカウントの確認

次の表は、Linux 仮想マシンにより使用されるアカウント名とアカウント タイプを示しています。

表 1-4. アカウント名およびアカウント タイプ

アカウント名	アカウントタイプ	使用
root	Linux OS に組み込み	Java スタンドアロン エージェント、mksvchanserver、シェル スクリプト
vmwblast	Linux Agent インストーラが作成	VMwareBlastServer
<現在のログイン ユーザー>	Linux OS に組み込み、Active Directory ユーザー、または LDAP ユーザー	Python スクリプト

デスクトップ環境

Horizon 7 for Linux デスクトップは、Ubuntu、SUSE、RHEL、および CentOS ディストリビューションのデフォルトのデスクトップ環境である GNOME デスクトップ環境のみをサポートします。

KDE デスクトップ環境をインストールしていると、Horizon 7 for Linux のインストーラは起動に失敗します。

2D または vSGA グラフィックスの仮想マシン設定

特定の Horizon 7 for Linux の仮想マシンを作成する場合、メモリ設定と構成パラメータを推奨される最小値に変更する必要があります。

NVIDIA vDGA を使用するように構成された仮想マシンでは、NVIDIA の物理グラフィック カードを使用します。NVIDIA GRID vGPU を使用するように構成されている仮想マシンでは、NVIDIA の物理グラフィック アクセラレータをベースとする NVIDIA 仮想グラフィック カードを使用します。これらの仮想マシンのビデオ メモリ (vRAM) 設定および構成パラメータを変更する必要はありません。

2D または vSGA グラフィックスを使用するように構成された仮想マシンでは、VMware 仮想グラフィック カードを使用します。また、これらのタイプの仮想マシンでは次の設定を変更する必要があります。

- ビデオ メモリ (vRAM) 設定
- 構成パラメータ
- 3D メモリ設定
- パフォーマンス要件を満たすための vCPU および仮想メモリ設定

[ビデオ メモリ (vRAM) 設定]

vSphere Client で Linux 仮想マシンを作成するときには、[表 1-5. 2D または vSGA グラフィックスで推奨される vRAM 設定](#)に示されているように vRAM サイズを構成します。仮想マシンで構成するモニタの数と解像度別に推奨される vRAM サイズを設定します。

表 1-5. 2D または vSGA グラフィックスで推奨される vRAM 設定

VRAM サイズ	モニタ数	最大解像度
10 MB	1	1600x1200 または 1680x1050
12 MB	1	1920x1440
32 MB	1	2560x1600
48 MB	2	2048x1536

VRAM サイズ	モニタ数	最大解像度
80 MB	2	2560x1600
128 MB	3	2560x1600
128 MB	4	2560x1600

これらの vRAM サイズは推奨される最小値です。仮想マシンでさらにリソースを利用できる場合は、ビデオ パフォーマンスを向上するために vRAM にさらに大きな値に設定します。

最低解像度の 1 つのモニタで構成されているマシンの推奨 vRAM 最小サイズは 10 MB です。

[仮想マシンを作成して、Linux インストールする](#)で説明されているように、ディスプレイの数と使用するビデオ メモリの量を設定するには、仮想マシンをパワーオフする必要があります。

Horizon 接続サーバ 7 は、Windows 仮想マシンの場合と同じように、Linux 仮想マシンでは vRAM 設定を自動的に構成しません。vSphere Client で vRAM 設定を手動で構成する必要があります。

Linux 仮想マシンが推奨サイズよりも小さい vRAM で構成されている場合、次の問題が発生する可能性があります。

- 最初の接続の直後にデスクトップ セッションが切断される。
- 自動調整が機能せず、画面の小さい領域にデスクトップが表示される。

Linux 仮想マシンの [ディスプレイの数] の値が実際に必要な数よりも少ない場合、1 台以上のモニタでデスクトップが空白で表示されます。

推奨された設定で自動調整の問題が発生する場合、さらに大きな vRAM サイズを指定することができます。vSphere Client では、最大で 128 MB の vRAM サイズが許可されます。128 MB を超過するサイズを指定した場合、vmx 構成ファイルを手動で変更する必要があります。次の例では、256 MB の vRAM サイズを指定しています。

```
svga.vramSize = "268435456"
```

[構成パラメータ]

複数のモニタに Linux リモート デスクトップを表示するには、仮想マシンのいくつかの構成パラメータを設定する必要があります。仮想マシンの構成パラメータを設定する一般的な手順は、次のとおりです。

- 1 仮想マシンをパワーオフします。
- 2 vSphere Web Client で仮想マシンを右クリックして、[設定の編集] を選択します。
- 3 [仮想マシン オプション] タブをクリックし、[詳細] をクリックします。
- 4 [設定の編集] をクリックしてから、[行を追加] をクリックします。
- 5 構成パラメータの名前と値を入力します。
- 6 [OK] をクリックして変更内容を保存します。

次の構成パラメータを設定する必要があります。

- `svga.autodetect` を `false` に設定します。

```
svga.autodetect="false"
```

- ディスプレイ モニタの数と向き（横または縦）に応じて、`svga.maxWidth` および `svga.maxHeight` の値を計算します。原則では、すべてのディスプレイをサポートするために `svga.maxWidth` 値と `svga.maxHeight` 値を十分な大きさにする必要があります。たとえば、最大解像度 2560x1600 で 4 台のディスプレイをサポートするには、次の値を設定する必要があります。

```
svga.maxHeight="3200"
svga.maxWidth="10240"
```

- vSphere 6.0 以降で動作する RHEL 6.8、CentOS 6.8、および Ubuntu 16.04 の場合、2D および vSGA の最大画面サイズは 4096x4096 になります。8192x8192 の画面サイズを使用するには、次のオプションを設定します。

```
mks.enable3d = TRUE
```

- vSphere 5.5 U3 を実行している RHEL 6.8、CentOS 6.8、および SLED 11 SP3/SP4 の場合は、次のオプションを設定します。他のディストリビューションではこのオプションは設定しないでください。設定すると、仮想マシンは起動できなくなります。

```
svga.capabilitiesMask="0xFF7FFFFFFF"
```

複数のモニタを使用する場合は、これらの構成パラメータを設定する必要があります。これらのパラメータを設定しないと、次の問題が 1 つ以上発生する可能性があります。

- デスクトップはいくつかのモニタに表示され、他のモニタには何も表示されません。
- キーストロークが何回も表示される。
- デスクトップが非常に遅くなる。
- 画面の小さい領域にデスクトップが表示される。

vCPU とメモリ設定

2D または vSGA デスクトップのパフォーマンスを向上させるには、Linux 仮想マシン用で vCPU と仮想メモリを増設します。たとえば、2 つの vCPU と 2GB の仮想メモリを設定します。

4 台のモニタがある場合など、複数のモニタを使用して大画面にする場合、仮想マシンに 4 つの vCPU と 4GB の仮想メモリを設定します。

2D または vSGA デスクトップでビデオを再生する場合、仮想マシンに 4 つの vCPU と 4GB の仮想メモリを設定します。

3D メモリ設定

vSGA の複数モニタ環境でパフォーマンスを向上させるには、仮想マシンの [3D メモリ] 設定を 1GB 以上に設定します。

デスクトップ デプロイのための Linux 仮想マシンの準備

2

Linux デスクトップのセットアップには、Linux 仮想マシンの作成およびリモート デスクトップ デプロイのためのオペレーティング システムの準備が含まれます。

この章には、次のトピックが含まれています。

- 仮想マシンを作成して、Linux インストールする
- リモート デスクトップ デプロイ用の Linux マシンの準備
- Horizon Agent 用依存パッケージのインストール

仮想マシンを作成して、Linux インストールする

Horizon 7 にデプロイする各リモート デスクトップに対して vCenter Server で新しい仮想マシンを作成します。仮想マシンに Linux ディストリビューションをインストールする必要があります。

前提条件

- デプロイする環境がサポートする Linux デスクトップの要件を満たしていることを確認します。 [Horizon 7 for Linux のシステム要件](#)を参照してください。
- vCenter Server で仮想マシンを作成し、ゲスト OS をインストールする手順について理解しておきます。View でのデスクトップ プールとアプリケーション プールの設定 ドキュメントの「仮想マシンの作成および準備」を参照してください。
- 仮想マシンで使用するモニタについて推奨されるビデオ メモリ (vRAM) の値を理解しておきます。 [Horizon 7 for Linux のシステム要件](#)を参照してください。

手順

- 1 vSphere Web Client または vSphere Client で新しい仮想マシンを作成します。

2 カスタム構成オプションを構成します。

- a 仮想マシンを右クリックし、[設定の編集] をクリックします。
- b vCPU の数と vMemory のサイズを指定します。

推奨値については、お使いの Linux ディストリビューションのインストール ガイドのガイドラインに従ってください。

たとえば、Ubuntu 12.04 では、2048 MB の vMemory と 2 台の vCPU を構成することが推奨されています。

- c [ビデオ カード] を選択して、ディスプレイの数とビデオ メモリ (vRAM) の合計を指定します。

VMware のドライバを使用し、2D や vSGA を使用する仮想マシンについては、vSphere Web Client で vRAM のサイズを設定します。vRAM のサイズは、NVIDIA のドライバを使用する vDGA や NVIDIA GRID vGPU マシンには影響しません。

推奨値については、Horizon 7 for Linux のシステム要件のガイドラインに従ってください。ビデオ メモリ 計算ツールは使用しないでください。

3 仮想マシンをパワーオンして、Linux ディストリビューションをインストールします。

4 GNOME デスクトップ環境としてのみ仮想マシンを構成します。

Ubuntu、SUSE、RHEL、および CentOS ディストリビューションをインストールする場合、GNOME デスクトップ環境がデフォルトの選択肢になります。KDE デスクトップ環境は選択しないでください。

5 システムのホスト名が 127.0.0.1 に対して解決可能であることを確認してください。

リモート デスクトップ デプロイ用の Linux マシンの準備

Horizon 7 をデプロイした環境でデスクトップとして使用するために Linux マシンを準備するには、特定のタスクを実行する必要があります。

Linux マシンを Horizon 7 で管理できるようにするには、マシンが接続サーバと通信する必要があります。Linux マシンが完全修飾ドメイン名 (FQDN) を使用して接続サーバ インスタンスに ping を送信できるように、Linux マシンのネットワークを構成する必要があります。

Open VMware Tools (OVT) は、RHEL 7、CentOS 7、および SLES 12 のマシンに事前にインストールされています。リモート デスクトップとして使用できるように、これらのマシンのいずれかを準備する場合は、下記の手順 1 ～ 5 をスキップできます。これらの手順には、手動でインストーラを実行して VMware Tools をインストールする方法が記載されています。

Ubuntu 16.04 マシンを使用している場合は、OVT をインストールします。このマシンをリモート デスクトップとして使用する場合は、下記の手順 1 ～ 5 をスキップでき、次のコマンドを使用して Ubuntu 16.04 マシンに OVT を手動でインストールできます。

```
apt-get install open-vm-tools-desktop
```


前提条件

- 新しい仮想マシン (VM) が vCenter Server で作成され、Linux ディストリビューションがマシンにインストールされていることを確認します。
- Linux 仮想マシンへの VMware Tools のマウントとインストールの手順を理解しておきます。vSphere 仮想マシン管理ドキュメントにある「Linux 仮想マシンでの VMware Tools の手動インストールまたはアップグレード」を参照してください。
- Linux マシンが DNS を介して解決できるように構成する手順を理解しておきます。これらの手順は、Linux ディストリビューションとリリースによって異なります。手順については、Linux ディストリビューションとリリースのドキュメントを参照してください。

手順

- 1 vSphere Web Client または vSphere Client で、VMware Tools 仮想ディスクを仮想マシンにマウントします。
- 2 VMware Tools のインストーラ ファイル `VMwareTools-x.x.x-xxxx.tar.gz` を右クリックして、[Extract to (展開先)] をクリックして、Linux ディストリビューションのデスクトップを選択します。

`vmware-tools-distrib` フォルダがデスクトップに展開されます。

- 3 仮想マシンで、root としてログインし、ターミナル ウィンドウを開きます。
- 4 VMware Tools の tar 形式のインストーラ ファイルを解凍します。

例：

```
tar xzpf /mnt/cdrom/VMwareTools-x.x.x-yyyy.tar.gz
```

- 5 インストーラを実行して VMware Tools を構成します。

Linux ディストリビューションによってこのコマンドは若干異なる場合があります。例：

```
cd vmware-tools-distrib
sudo ./vmware-install.pl -d
```

通常、インストーラ ファイルの実行が終了した後に、`vmware-config-tools.pl` 構成ファイルが実行されます。

- 6 Linux マシンのホスト名を `/etc/hosts` ファイルの 127.0.0.1 にマッピングします。

RHEL、CentOS、SLES、SLED の場合、ホスト名は自動的にマッピングされないため、127.0.0.1 に手動でマッピングする必要があります。Ubuntu の場合、デフォルトでマッピングされるため、この手順は不要です。この手順はデスクトップを一括デプロイする場合も不要です。このマッピングがクローン作成プロセスによって追加されるためです。

注： Horizon Agent をインストールした後に Linux マシンのホスト名を変更する場合は、新しいホスト名を `/etc/hosts` ファイルの 127.0.0.1 にマッピングする必要があります。マッピングしないと、古いホスト名が引き続き使用されます。

- 7 RHEL 7 および CentOS 7 については、`virbr0` が無効になっていることを確認します。

```
virsh net-destroy default
virsh net-undefine default
service libvirtd restart
```

- 8 ポッドにある View 接続サーバーインスタンスを DNS を介して解決できることを確認します。
- 9 デフォルトのランレベルが 5 になるように Linux マシンを構成します。
- Linux デスクトップが機能するには、ランレベルを 5 にする必要があります。
- 10 OpenLDAP サーバを使用して認証するように構成された Ubuntu マシンに、マシンで完全修飾ドメイン名を設定します。

この手順によって、View Administrator の [セッション] ページの [ユーザー] フィールドにこの情報を正しく表示できるようになります。/etc/hosts ファイルを次のように編集します。

a # nano /etc/hosts

b 完全修飾ドメイン名を追加します。たとえば、127.0.0.1 hostname.domainname hostname のように追加します。

c 終了してファイルを保存します。

- 11 SUSE については、[DHCP 経由でホスト名を変更] を無効にします。ホスト名またはドメイン名を設定します。

a Yast では、[ネットワーク設定] をクリックします。

b [ホスト名/DNS] タブをクリックします。

c [DHCP 経由でホスト名を変更] の選択を解除します。

d ホスト名とドメイン名を入力します。

e [OK] をクリックします。

VMware Tools をインストールした後に、Linux カーネルをアップグレードすると、VMware Tools の実行が停止する場合があります。この問題を解決するには、<http://kb.vmware.com/kb/2050592> を参照してください。

Horizon Agent 用依存パッケージのインストール

Linux 版 Horizon Agent には、Linux ディストリビューションに一意の依存パッケージがあります。Linux 版 Horizon Agent をインストールする前に、これらのパッケージをインストールする必要があります。

前提条件

新しい仮想マシン (VM) が vCenter Server で作成され、Linux ディストリビューションがマシンにインストールされていることを確認します。

手順

- 1 デフォルトではインストールまたはアップグレードされない必須パッケージをインストールします。パッケージが要件を満たさない場合、インストーラはインストールを中断します。

表 2-1. 必須の依存パッケージ

Linux ディストリビューション	パッケージ
RHEL 7 または CentOS 7 2D デスクトップおよび vSGA デスクトップ用に VMware グラフィックス ドライバをアップグレードします。 注: vDGA および vGPU は、仮想マシンで NVIDIA グラフィックス ドライバを使用するため、このオプションが適用されません。	<pre>yum install mesa-libxatracker xorg-x11-drv-vmware mesa-private-llvm mesa-dri-drivers</pre> <p>アップデートすると、次のバージョン以降がパッケージに関連付けられます。</p> <ul style="list-style-type: none"> ■ xorg-x11-drv-vmware-13.0.2-7.20150211git8f0cf7c.el7.x86_64 ■ mesa-libxatracker-10.6.5-3.20150824.el7.x86_64 ■ mesa-private-llvm-3.6.2-2.el7.x86_64 ■ mesa-dri-drivers-10.6.5-3.20150824.el7.x86_64 <p>注: この更新は、CentOS のデフォルトのオンライン リポジトリ向けには使用できません。CentOS Continuous Release (CR) リポジトリを有効にする必要があります。</p> <pre>yum-config-manager --enable "CentOS-7 - cr"</pre> <p>VMware グラフィックス ドライバをアップグレードした後は、リポジトリを再び無効にすることができます。</p> <pre>yum-config-manager --disable "CentOS-7 - cr"</pre>
SLED 11 SP3/SP4 xorg-x11-server を 7.4.27.111.1 より後のバージョンにアップグレードします	<pre>zypper install xorg-x11-server</pre>
SLES 12 SP1 (Horizon 7 バージョン 7.0.1/7.0.2 の場合) VMware グラフィックス ドライバ xf86-video-vmware を 13.1.0-5.2 より後のバージョンにアップグレードします	<p>インストールするには、OpenSUSE オンライン リポジトリを有効にする必要があります。</p> <ol style="list-style-type: none"> 1 <pre>zypper addrepo http://download.opensuse.org/distribution/leap/42.1/repo/oss/ oss-42.1</pre> 2 <pre>zypper ref</pre> 3 <pre>zypper install xf86-video-vmware</pre> 4 <pre>zypper removerepo oss-42.1</pre>
SLES 12 SP1 (Horizon 7 バージョン 7.0.3 以降の場合) SUSE リポジトリで xf86-video-vmware を 13.0.2-3.2 以降のバージョンにアップグレードします	<ol style="list-style-type: none"> 1 SUSE 12 を登録して SUSE リポジトリを有効にします。 <pre>SUSEConnect -r <i>Registration Code</i> -e <i>Email</i></pre> 2 xf86-video-vmware のバージョンを更新します。 <pre>zypper install xf86-video-vmware</pre>

Linux ディストリビューション	パッケージ
SLES 12 SP1	<p>Horizon Agent をインストールする場合、SLES 12 Linux デスクトップに python-gobject2 をインストールする必要があります。</p> <ol style="list-style-type: none"> 1 OpenSUSE リポジトリを削除します。 2 SUSE 12 を登録して SUSE リポジトリを有効にします。 <pre>SUSEConnect -r <i>Registration Code</i> -e <i>Email</i></pre> <ol style="list-style-type: none"> 3 python-gobject2 をインストールします。 <pre>zypper install python-gobject2</pre> <p>注: OpenSUSE 上の python-object2 は、SUSE 12 と互換性がありません。</p>
Ubuntu 14.04 indicator-session を、 https://launchpad.net/ubuntu/wily/amd64/indicator-session/12.10.5+15.04.20150327-0ubuntu1 1 で利用可能な 12.10.5+15.04.20150327 にアップ グレードします。	<pre>wget http://launchpadlibrarian.net/201393830/indicator-session_12.10.5+15.04.20150327-0ubuntu1_amd64.deb</pre> <pre>sudo dpkg -i ./indicator-session_12.10.5+15.04.20150327-0ubuntu1_amd64.deb</pre>
Ubuntu 16.04	<pre>apt-get install python-dbus python-gobject</pre>

2 Horizon Agent のオプション パッケージをインストールします。

- デフォルトでは、RHEL または CentOS 6.7 は、デッドロック問題を引き起こす可能性のある glibc-2.12-1.166.el6.x86_64 をインストールします。その結果、デスクトップ接続は停止します。この問題を解決するには、オンライン リポジトリで glibc を最新バージョンにアップグレードする必要があります。

```
sudo yum install glibc
```

- 複数のモニタがある Ubuntu 14.04 デスクトップでは、パフォーマンスを向上させる目的で Compiz を無効にするために gnome-session-fallback が必要となります。

```
sudo apt-get install gnome-session-fallback
```

- 複数のモニタがある Ubuntu 16.04 デスクトップでは、パフォーマンスを向上させる目的で Compiz を無効にするために gnome-session-flashback が必要となります。

```
sudo apt-get install gnome-session-flashback
```

Linux デスクトップの Active Directory 統合のセットアップ

3

View は、ユーザーの認証と管理に既存の Microsoft Active Directory (AD) インフラストラクチャを使用します。Linux デスクトップを Active Directory と統合すると、ユーザーは Active Directory ユーザー アカウントを使用して Linux デスクトップにログインできるようになります。

この章には、次のトピックが含まれています。

- [Linux と Active Directory の統合](#)
- [シングル サインオンとスマート カード リダイレクトのセットアップ](#)

Linux と Active Directory の統合

Linux と Active Directory (AD) を統合するためのソリューションは複数あります。

次のソリューションは View 環境で動作することが分かっています。

- OpenLDAP パススルー認証
- Winbind

OpenLDAP パススルー認証ソリューションには、おおまかに次のような手順が含まれます。

- パスワード検証を別のプロセス (saslauthd など) に委任するように OpenLDAP サーバを構成します。saslauthd は Active Directory に対してパスワード検証を実行できます。
- OpenLDAP でユーザーを認証するように Linux デスクトップを構成します。

Linux デスクトップを一括デプロイする予定がある場合は、最後の Active Directory 統合タスクを実行するようにテンプレート仮想マシン (VM) をセットアップできます。次の考慮事項を認識しておく必要があります。

- OpenLDAP ソリューションは追加の手順なしでクローン仮想マシンで動作します。
- Winbind ソリューションでは、各クローン仮想マシンには別のホスト名があるため、ドメインに参加する手順が失敗します。各クローン仮想マシンは、次のコマンドを実行してドメインに再参加する必要があります。

```
sudo /usr/bin/net ads join -U <domain user>%<domain password>
```

winbind ソリューション用のクローンが作成された仮想マシンで、ドメイン参加コマンドを実行するには、次のオプションを使用します。

- SSH または vSphere PowerCLI などの各仮想マシンにリモート接続してコマンドを実行します。スクリプトの詳細については、[8 章 手動デスクトップ プールのための Horizon 7 の一括デプロイ](#) を参照してください。

- コマンドをシェル スクリプトに含めて、スクリプト パスを `/etc/vmware/viewagent-custom.conf` の Horizon Agent オプション `RunOnceScript` に指定します。詳細については、[Linux デスクトップでの構成ファイルのオプション設定](#)を参照してください。

Linux デスクトップの一括デプロイに関する詳細については、[8 章 手動デスクトップ プールのための Horizon 7 の一括デプロイ](#)を参照してください。

シングル サインオンとスマート カード リダイレクトのセットアップ

シングル サインオン (SSO) とスマート カード リダイレクトをセットアップするには、構成手順をいくつか実行する必要があります。

シングル サインオン

Horizon View のシングル サインオン モジュールは Linux で PAM (プラグ可能な認証モジュール) と対話し、Linux を Active Directory (AD) と統合するために使用する方法には依存しません。Horizon View の SSO は、Linux を Active Directory と統合する OpenLDAP と WinBind のソリューションで動作することが知られています。

デフォルトの場合、SSO では、Active Directory の `sAMAccountName` 属性がログイン ID であると想定されます。OpenLDAP か Winbind ソリューションを使用する場合、正しいログイン ID を SSO に使用するには、次の構成手順を実行する必要があります。

- OpenLDAP では、`sAMAccountName` を `uid` に設定します。
- Winbind では、次のステートメントを構成ファイル `/etc/samba/smb.conf` に追加します。

```
winbind use default domain = true
```

ユーザーがドメイン名を指定してログインする必要がある場合は、`SSOUserFormat` オプションを Linux デスクトップで設定する必要があります。詳細については、[Linux デスクトップでの構成ファイルのオプション設定](#)を参照してください。SSO では常に、大文字で短いドメイン名が使用されることに注意してください。たとえば、ドメインが `mydomain.com` である場合、SSO では `MYDOMAIN` がドメイン名として使用されます。このため、`SSOUserFormat` オプションを設定するときには、`MYDOMAIN` を指定する必要があります。短いドメイン名と長いドメイン名については、次のルールが適用されます。

- OpenLDAP では、大文字で短いドメイン名を使用する必要があります。
- Winbind では、長いドメイン名と短いドメイン名が両方ともサポートされます。

Active Directory ではログイン名で特殊文字がサポートされますが、Linux ではサポートされません。このため、SSO のセットアップ時には、特殊文字をログイン名に使用しないでください。

Active Directory では、ユーザーの `UserPrincipalName` (UPN) 属性と `sAMAccount` 属性が一致せずに、ユーザーが UPN でログインすると、SSO は失敗します。ユーザーが、`sAMAccount` に保存されている名前を使用してログインすると、これを回避できます。

View では、ユーザー名の大文字と小文字を区別する必要はありません。Linux オペレーティング システムで、大文字と小文字を区別しないユーザー名を処理できることを確認してください。

- Winbind では、ユーザー名の大文字と小文字がデフォルトで区別されません。

- OpenLDAP では、Ubuntu が NSCD を使用してユーザーを認証し、大文字と小文字がデフォルトで区別されません。RHEL と CentOS は SSSD を使用してユーザーを認証し、大文字と小文字がデフォルトで区別されます。設定を変更するには、ファイル `/etc/sss/sss.conf` を編集して次の行を `[domain/default]` セクションに追加します。

```
case_sensitive = false
```

スマート カード リダイレクト

スマート カード リダイレクトをセットアップするには、最初に Linux ディストリビュータとスマート カード ベンダーの指示に従ってください。次に、`pcsc-lite` パッケージを 1.7.4 に更新します。たとえば、次のコマンドを実行します。

```
#yum groupinstall "Development tools"
#yum install libudev-devel
#service pcscd stop
#wget https://alioth.debian.org/frs/download.php/file/3598/pcsc-lite-1.7.4.tar.bz2
#tar -xjvf pcsc-lite-1.7.4.tar.bz2
#cd ./pcsc-lite-1.7.4
#./configure --prefix=/usr/ --libdir=/usr/lib64/ --enable-usbdropdir=/usr/lib64/pcsc/drivers
--enable-confdir=/etc --enable-ipcdire=/var/run --disable-libusb --disable-serial --disable-usb
--disable-libudev
#make
#make install
#service pcscd start
```

Winbind では、次のステートメントを構成ファイル `/etc/samba/smb.conf` に追加します。

```
winbind use default domain = true
```

Horizon Agent をインストールするときは、最初に SELinux を無効にするか、SELinux の permissive モードを有効にする必要があります。スマート カード リダイレクトのコンポーネントを明確に選択する必要もあります。コンポーネントはデフォルトでは選択されません。詳細については、[install_viewagent.sh コマンドライン オプション](#) を参照してください。

スマートカード SSO は、Horizon View 7.0.1 以降で有効です。スマート カード リダイレクト機能を仮想マシンにインストールすると、vSphere Client の USB リダイレクトはスマート カードで動作しません。

スマート カード リダイレクトでは、1 つのスマート カード リーダのみがサポートされます。複数のリーダをクライアント デバイスに接続すると、この機能は動作しません。

スマート カード リダイレクトでは、カードで 1 つの証明書のみがサポートされます。複数の証明書がカードに存在する場合、最初のスロットの証明書が使用され、その他の証明書は無視されます。これは Linux の制限です。

注:

- スマートカードは、次の winbind 値をサポートします。それ以外の場合は、スマートカード SSO と手動のログインは失敗します。

```
winbind use default domain=true
```

- PIV カード (Linux デスクトップ スマートカード リダイレクトでサポート) でブローカを認証するために Linux クライアントを使用する場合、SSL エラーを回避するために、Linux クライアント構成用に `~/.vmware/view-preferences` にて `view.sslProtocolString = "TLSv1.1"` を追加する必要があります。
-

Linux デスクトップのグラフィックス のセットアップ

4

ESXi ホストまたはゲスト OS で NVIDIA 機能を活用するように、RHEL 6.6/6.7/6.8 および 7.2 を構成できます。

[3D グラフィックスのセットアップのための仮想マシンのクローン作成要件]

3D グラフィックスをセットアップする前に、仮想マシンのクローン作成に関する次の要件を考慮する必要があります。

- vGPU および vSGA については、基本仮想マシンのグラフィックスのセットアップを完了させます。仮想マシンのクローンを作成します。グラフィックス設定はクローン作成された仮想マシンについて動作し、さらなる設定は必要ありません。
- vDGA については、基本仮想マシンのグラフィックスのセットアップを完了させます。仮想マシンのクローンを作成します。ただし、クローン作成された仮想マシンをパワーオンする前に、クローン作成された仮想マシンから既存の NVIDIA パススルー PCI デバイスを削除し、クローン作成された仮想マシンに新しい NVIDIA パススルー PCI デバイスを追加する必要があります。NVIDIA パススルー PCI デバイスは、仮想マシン間で共有することはできません。各仮想マシンは、専用の NVIDIA パススルー PCI デバイスを使用します。

この章には、次のトピックが含まれています。

- [vGPU を使用するための RHEL 6.6/6.7/6.8 と RHEL 7.2 の構成](#)
- [vDGA を使用するための RHEL 6.6/6.7/6.8 の構成](#)
- [vSGA を使用するための RHEL 7.2 の構成](#)

vGPU を使用するための RHEL 6.6/6.7/6.8 と RHEL 7.2 の構成

RHEL 6.6/6.7/6.8 と RHEL 7.2 をセットアップすると、NVIDIA vGPU（共有 GPU ハードウェア アクセラレーション）機能を ESXi ホストで利用できます。

ESXi ホスト GPU ドライバ (.vib) と一致する NVIDIA Linux 仮想マシン ディスプレイ ドライバを使用する必要があります。ドライバパッケージに関する情報は、NVIDIA の Web サイトを参照してください。

重要: NVIDIA vGPU は、NVIDIA Maxwell M60 グラフィックス カードと NVIDIA M6 グラフィックス カードでサポートされます。この機能は、GRID K1 や K2 などの他の NVIDIA グラフィックス カードでは動作しません。

注意: 開始する前に、Horizon Agent が Linux 仮想マシンにインストールされていないことを確認します。NVIDIA vGPU を使用するようにマシンを構成する前に Horizon Agent をインストールすると、`xorg.conf` ファイルで必須の構成パラメータが上書きされ、NVIDIA vGPU は動作しません。NVIDIA vGPU の構成が完了した後に、Horizon Agent をインストールする必要があります。

NVIDIA GRID vGPU グラフィック カードの VIB の ESXi ホストへのインストール

ESXi 6.0 U1 以降のホストに NVIDIA GRID グラフィック カードの VIB をダウンロードしてインストールする必要があります。

NVIDIA から vGPU Manager を含む vGPU ソフトウェア パッケージと Linux ディスプレイ ドライバが提供されます。vGPU ソフトウェア パッケージは、この手順で ESXi ホストにインストールし、Linux ディスプレイ ドライバは、この後の手順で Linux 仮想マシンにインストールします。

前提条件

- vSphere 6.0 U1 以降のリリースが環境にインストールされていることを確認します。
- NVIDIA Maxwell M60 GPU または M6 GPU が ESXi ホストにインストールされていることを確認します。

手順

- 1 [NVIDIA のドライバ ダウンロード](#) サイトから NVIDIA GRID vGPU グラフィック カードの VIB をダウンロードします。

適切な VIB バージョンをドロップダウン メニューから選択します。

オプション	説明
製品タイプ	[GRID]
製品シリーズ	[NVIDIA GRID vGPU] を選択します。
製品	ESXi ホストにインストールされるバージョン ([GRID K2] など) を選択します。
オペレーティング システム	VMware vSphere ESXi のバージョンを選択します。

- 2 vGPU ソフトウェア パッケージの .zip ファイルを解凍します。
- 3 vGPU Manager フォルダを ESXi 6.0 U1 ホストにアップロードします。

注: この後の手順で、Linux ディスプレイ ドライバを Linux 仮想マシンにインストールします。

- 4 ESXi ホスト上のすべての仮想マシンをパワーオフまたはサスペンドします。
- 5 SSH を使用して ESXi ホストに接続します。
- 6 xorg サービスを停止します。

```
# /etc/init.d/xorg stop
```

- 7 NVIDIA VIB をインストールします。

例 :

```
# esxcli system maintenanceMode set --enable true
# esxcli software vib install -v /path-to-vib/NVIDIA-VIB-name.vib
# esxcli system maintenanceMode set --enable false
```

8 ESXi ホストを再起動または更新します。

- ◆ インストール済みの ESXi ホストでは、ホストを再起動します。
- ◆ ステートレス ESXi ホストでは、次の手順を実行し、ホストを更新します（これらの手順は、インストール済みのホストにも適用できます）。

```
Update vmkdevmgr:
# kill -HUP $(cat /var/run/vmware/vmkdevmgr.pid)

Wait for the update to complete:
# localcli --plugin-dir /usr/lib/vmware/esxcli/int deviceInternal bind

This is a new requirement with the NVIDIA 352.* host driver:
# /etc/init.d/nvidia-vgpu start

Restart xorg, which is used for GPU assignment:
# /etc/init.d/xorg start
```

9 ホストを再起動した後に、xorg サービスが実行されていることを確認します。

Linux 仮想マシンで vGPU を使用するための共有 PCI デバイスの構成

NVIDIA vGPU を使用するには、Linux 仮想マシン用に共有 PCI デバイスを構成する必要があります。

前提条件

- Linux 仮想マシンをデスクトップとして使用する準備ができていることを確認します。[仮想マシンを作成して、Linux インストールするおよびリモート デスクトップ デプロイ用の Linux マシンの準備](#)を参照してください。
- Horizon Agent が Linux 仮想マシンにインストールされていないことを確認します。
- NVIDIA VIB が ESXi ホストにインストールされていることを確認します。[NVIDIA GRID vGPU グラフィック カードの VIB の ESXi ホストへのインストール](#)を参照してください。
- NVIDIA vGPU で利用可能な仮想 GPU タイプについて理解しておきます。GPU のタイプは、[GPU プロファイル] 設定で選択します。仮想 GPU タイプは、ESXi ホストにインストールされた物理 GPU でさまざまな機能を提供します。[NVIDIA 仮想 GPU タイプ](#)を参照してください。

手順

- 1 仮想マシンをパワーオフします。
- 2 vSphere Web Client で、[仮想マシン ハードウェア] タブで仮想マシンを選択して、[設定編集] をクリックします。
- 3 [新規デバイス] メニューで、[共有 PCI デバイス] を選択します。
- 4 [追加] をクリックして、ドロップダウン メニューから [NVIDIA GRID vGPU] を選択します。
- 5 [GPU プロファイル] 設定で、ドロップダウン メニューから仮想 GPU タイプを選択します。
- 6 [すべてのメモリを予約] をクリックして、[OK] をクリックします。

GPU が NVIDIA GRID vGPU をサポートできるようにするには、すべての仮想マシンのメモリを予約する必要があります。

7 仮想マシンをパワーオンします。

NVIDIA 仮想 GPU タイプ

vSphere Web Client の [仮想ハードウェア] ページにある [GPU プロファイル] 設定で、ESXi ホストの物理 NVIDIA GPU で特別な機能を利用できるようにする仮想 GPU タイプを選択できます。

Linux 仮想マシンでは、NVIDIA GRID vGPU は、NVIDIA Maxwell M60 GPU または NVIDIA M6 GPU でサポートされます。

表 4-1. Linux 仮想マシンの NVIDIA GRID M60 vGPU で利用可能な仮想 GPU タイプ

仮想 GPU タイプ	物理ボード	物理 GPU の数	仮想 GPU あたりの FB	ディスプレイ数	最大解像度	物理 GPU あたりの最大仮想 GPU 数	物理ボードあたりの最大仮想 GPU 数
GRID M60-0q	GRID M60	2	512M	2	2560x1600	16	32
GRID M60-1q	GRID M60	2	1G	2	2560x1600	8	16
GRID M60-2q	GRID M60	2	2G	4	2560x1600	4	8
GRID M60-4q	GRID M60	2	4G	4	3840x2160	2	4
GRID M60-8q	GRID M60	2	8G	4	3840x2160	1	2

表 4-2. Linux 仮想マシンの NVIDIA GRID M6 vGPU で利用可能な仮想 GPU タイプ

仮想 GPU タイプ	物理ボード	物理 GPU の数	仮想 GPU あたりの FB	ディスプレイ数	最大解像度	物理 GPU あたりの最大仮想 GPU 数	物理ボードあたりの最大仮想 GPU 数
GRID M6-0q	GRID M6	1	512M	2	2560x1600	16	16
GRID M6-1q	GRID M6	1	1G	2	2560x1600	8	8
GRID M6-2q	GRID M6	1	2G	4	2560x1600	4	4
GRID M6-4q	GRID M6	1	4G	4	3840x2160	2	2
GRID M6-8q	GRID M6	1	8G	4	3840x2160	1	1

NVIDIA GRID vGPU ディスプレイ ドライバのインストール

NVIDIA GRID vGPU ディスプレイ ドライバをインストールするには、デフォルトの NVIDIA ドライバを無効にし、NVIDIA ディスプレイ ドライバをダウンロードして、仮想マシンで PCI デバイスを構成する必要があります。

前提条件

- NVIDIA のダウンロード サイトから vGPU ソフトウェア パッケージをダウンロードして解凍しており、Linux ディスプレイ ドライバ (パッケージ コンポーネント) の準備ができていることを確認します。 [NVIDIA GRID vGPU グラフィック カードの VIB の ESXi ホストへのインストール](#) を参照してください。

また、共有 PCI デバイスが仮想マシンに追加されていることを確認します。 [Linux 仮想マシンで vGPU を使用するための共有 PCI デバイスの構成](#) を参照してください。

手順

- 1 デフォルトの NVIDIA Nouveau ドライバを無効にしてブラックリストに入れます。

- a `grub.conf` ファイルを編集します。

RHEL 6.6/6.7/6.8 の場合のファイルは `/boot/grub/grub.conf` です。RHEL 7.2 の場合のファイルは `/etc/default/grub.conf` です。

RHEL バージョン	コマンド
6.6/6.7/6.8	<code>sudo vi /boot/grub/grub.conf</code>
7.2	<code>sudo vi /etc/default/grub.conf</code>

- b `rdblacklist=nouveau` 行をカーネル オプションの最後に追加します。
- c `blacklist.conf` ファイルを編集します。

```
sudo vi /etc/modprobe.d/blacklist.conf
```

- d `blacklist.conf` ファイルの任意の場所に次の行を追加します。

```
blacklist nouveau
```

- 2 仮想マシンを再起動します。

表示のルック アンド フィールドが変更されます。

- 3 (オプション) Nouveau ドライバが無効になっていることを確認します。

```
/sbin/lsmmod | grep nouveau
```

`grep` 検索によって何も結果が返されない場合、Nouveau ドライバは無効になっています。

- 4 NVIDIA Linux ディスプレイ ドライバを仮想マシンにコピーします。
- 5 仮想マシンへのリモート ターミナルを開くか、`Ctrl + Alt + F2` キーを押してテキスト コンソールに切り替えて、`root` としてログインして、`init 3` コマンドを実行して X Windows を無効にします。
- 6 NVIDIA ドライバで必要となる追加のコンポーネントをインストールします。

```
sudo yum install gcc-c++
sudo yum install kernel-devel-$(uname -r)
sudo yum install kernel-headers-$(uname -r)
```

- 7 NVIDIA GRID vGPU ドライバ パッケージに実行可能なフラグを追加します。

```
chmod +x NVIDIA-Linux-x86_64-version-grid.run
```

8 NVIDIA GRID vGPU インストーラを起動します。

```
sudo ./NVIDIA-Linux-x86_64-version-grid.run
```

9 NVIDIA のソフトウェア使用許諾契約書に同意して、[Yes] を選択して、X の構成設定を自動的に更新します。

次のステップ

Linux 仮想マシンに Horizon Agent をインストールします。[Linux 仮想マシンへの Horizon Agent のインストール](#)を参照してください。

構成した Linux 仮想マシンを含むデスクトップ プールを作成します。[Linux 版手動デスクトップ プールの作成](#)を参照してください。

NVIDIA ディスプレイ ドライバがインストールされているかどうかの確認

View デスクトップ セッションに NVIDIA ドライバの出力を表示して、NVIDIA ディスプレイ ドライバが RHEL 6.6/6.7/6.8 仮想マシンにインストールされていることを確認できます。

前提条件

- NVIDIA ディスプレイ ドライバをインストールしていることを確認します。
- Horizon Agent が Linux 仮想マシンにインストールされていることを確認します。[Linux 仮想マシンへの Horizon Agent のインストール](#)を参照してください。
- Linux 仮想マシンがデスクトップ プールにデプロイされていることを確認します。[Linux 版手動デスクトップ プールの作成](#)を参照してください。

手順

1 Linux 仮想マシンを再起動します。

Horizon Agent 起動スクリプトは、X サーバを初期化し、トポロジを表示します。

vSphere コンソールで、仮想マシンの表示を参照することはできなくなります。

2 Horizon Client で、Linux デスクトップに接続します。

3 Linux デスクトップセッションで、NVIDIA ディスプレイ ドライバがインストールされていることを確認します。

ターミナル ウィンドウを開き、`glxinfo | grep NVIDIA` コマンドを実行します。

NVIDIA ドライバ出力が表示されます。例：

```
[root]# glxinfo | grep NVIDIA
server glx vendor string: NVIDIA Corporation
client glx vendor string: NVIDIA Corporation
OpenGL vendor string: NVIDIA Corporation
OpenGL version string: 4.5.0 NVIDIA 346.47
OpenGL shading language version string: 4.50 NVIDIA
```

ユーザーは、リモート デスクトップで NVIDIA グラフィックスの機能にアクセスできます。

NVIDIA ディスプレイ ドライバのインストールを確認した後、インストールが正しく動作するために、次のタスクを実行します。

- Linux カーネルをアップグレードする場合、Horizon Agent が View 接続サーバと通信できないことがあります。この問題を解決するには、NVIDIA ドライバを再インストールします。
- Linux 仮想マシンで NVIDIA GRID のライセンスを設定します。詳細については、NVIDIA のドキュメントを参照してください。ライセンスが設定されていない場合は、Linux デスクトップが正しく動作しません。たとえば、自動的に合わせる機能が動作しません。

vDGA を使用するための RHEL 6.6/6.7/6.8 の構成

Horizon 7 for Linux デスクトップが ESXi ホストで vDGA 機能を利用できるように、RHEL 6.6/6.7/6.8 ゲスト OS をセットアップできます。

注意: 開始する前に、Horizon Agent が Linux 仮想マシンにインストールされていないことを確認します。vDGA を使用するようにマシンを構成する前に Horizon Agent をインストールすると、`xorg.conf` ファイルで必須の構成パラメータが上書きされ、vDGA は動作しません。vDGA の構成が完了した後に、Horizon Agent をインストールする必要があります。

ホストで NVIDIA GRID を使用するために DirectPath I/O を有効にする

Linux 仮想マシンを構成して vDGA を使用できるようにするには、NVIDIA GRID GPU PCI デバイスを ESXi ホストの DirectPath I/O パススルーで利用できるようにする必要があります。

前提条件

- vSphere 6.0 または以降のリリースが環境にインストールされていることを確認します。
- NVIDIA GRID K1 または K2 グラフィック カードが ESXi ホストにインストールされていることを確認します。

手順

- 1 vSphere Web Client で、ESXi ホストを参照します。
- 2 [管理] タブをクリックして、[設定] をクリックします。
- 3 [ハードウェア] セクションの [PCI デバイス] をクリックします。
- 4 NVIDIA GRID GPU で DirectPath I/O パススルーを有効にするには、[編集] をクリックします。

アイコン	説明
緑色のアイコン	PCI デバイスはアクティブで、有効にできます。
オレンジ色のアイコン	デバイスの状態が変更されました。デバイスを使用する前にホストを再起動する必要があります。

- 5 NVIDIA GRID GPU を選択して、[OK] をクリックします。
PCI デバイスが表に追加され、仮想マシンで DirectPath I/O PCI デバイスを利用できるようになります。
- 6 ホストを再起動して、Linux 仮想マシンが PCI デバイスを利用できるようにします。

vDGA パススルー デバイスの RHEL 6.6/6.7/6.8 仮想マシンへの追加

vDGA を使用するように RHEL 6.6/6.7/6.8 仮想マシンを構成するには、PCI デバイスを仮想マシンに追加する必要があります。この手順によって、ESXi ホストの物理デバイスをパススルーして仮想マシンで使用できるようになります。

前提条件

- Linux 仮想マシンをデスクトップとして使用する準備ができていることを確認します。[仮想マシンを作成して、Linux インストールする](#)および[リモート デスクトップ デプロイ用の Linux マシンの準備](#)を参照してください。
- Horizon Agent が Linux 仮想マシンにインストールされていないことを確認します。
- NVIDIA GRID GPU PCI デバイスがホストの DirectPath I/O パススルーで利用可能になっていたか確認します。[ホストで NVIDIA GRID を使用するために DirectPath I/O を有効にする](#)を参照してください。

手順

- 1 sudo 権限で構成されたローカル ユーザーとして、RHEL 6.6/6.7/6.8 ゲスト OS にログインします。
- 2 vSphere Web Client で、[仮想マシン ハードウェア] タブで仮想マシンを選択して、[設定編集] をクリックします。
- 3 [新規デバイス] メニューで、[PCI デバイス] を選択します。
- 4 [追加] をクリックして、ドロップダウン メニューから PCI デバイスを選択します。
- 5 [すべてのメモリを予約] をクリックして、[OK] をクリックします。

GPU が vDGA をサポートできるようにするには、すべての仮想マシンのメモリを予約する必要があります。

- 6 仮想マシンをパワーオンして、vSphere コンソールを開いてマシンに接続します。
- 7 NVIDIA GRID デバイスが仮想マシンにパススルーされていることを確認します。

ターミナル ウィンドウを開き、次のコマンドを実行します。

```
lspci | grep NVIDIA
```

XX:00.0 VGA 互換のコントローラが表示されます。例：

```
NVIDIA Corporation GK104GL [GRID K2]
```

vDGA 用の NVIDIA ディスプレイ ドライバのインストール

vDGA 用の NVIDIA ディスプレイ ドライバをインストールするには、デフォルトの NVIDIA ドライバを無効にし、NVIDIA ディスプレイ ドライバをダウンロードして、仮想マシンで PCI デバイスを構成する必要があります。

前提条件

- PCI デバイスが RHEL 6.6/6.7/6.8 仮想マシンに追加されていることを確認します。[vDGA パススルー デバイスの RHEL 6.6/6.7/6.8 仮想マシンへの追加](#)を参照してください。

手順

- 1 デフォルトの NVIDIA Nouveau ドライバを無効にしてブラックリストに入れます。

- a `grub.conf` ファイルを編集します。

RHEL 6.6/6.7/6.8 の場合のファイルは `/boot/grub/grub.conf` です。RHEL 7.2 の場合のファイルは `/etc/default/grub.conf` です。

RHEL バージョン	コマンド
6.6/6.7/6.8	<code>sudo vi /boot/grub/grub.conf</code>
7.2	<code>sudo vi /etc/default/grub.conf</code>

- b `rdblacklist=nouveau` 行をカーネル オプションの最後に追加します。
- c `blacklist.conf` ファイルを編集します。

```
sudo vi /etc/modprobe.d/blacklist.conf
```

- d `blacklist.conf` ファイルの任意の場所に次の行を追加します。

```
blacklist nouveau
```

- 2 仮想マシンを再起動します。

表示のルック アンド フィールドが変更されます。

- 3 (オプション) Nouveau ドライバが無効になっていることを確認します。

```
/sbin/lsmmod | grep nouveau
```

`grep` 検索によって何も結果が返されない場合、Nouveau ドライバは無効になっています。

- 4 [NVIDIA のドライバ ダウンロード](#) サイトから NVIDIA のドライバをダウンロードします。

適切なドライバ バージョンを NVIDIA のドロップダウン メニューから選択します。

オプション	説明
製品タイプ	[GRID]
製品シリーズ	[GRID シリーズ]
製品	ESXi ホストにインストールされるバージョン ([GRID K2] など) を選択します。
オペレーティング システム	Linux 64 ビットまたは Linux 32 ビット

- 5 仮想マシンへのリモート ターミナルを開くか、`Ctrl + Alt + F2` キーを押してテキスト コンソールに切り替えて、`root` としてログインして、`init 3` コマンドを実行して X Windows を無効にします。

- 6 NVIDIA ドライバで必要となる追加のコンポーネントをインストールします。

```
sudo yum install gcc-c++  
sudo yum install kernel-devel-$(uname -r)  
sudo yum install kernel-headers-$(uname -r)
```

- 7 vDGA 用の NVIDIA ドライバ パッケージに実行可能なフラグを追加します。

```
chmod +x NVIDIA-Linux-x86_64-version.run
```

- 8 NVIDIA インストーラを起動します。

```
sudo ./NVIDIA-Linux-x86_64-version.run
```

- 9 NVIDIA のソフトウェア使用許諾契約書に同意して、[Yes] を選択して、X の構成設定を自動的に更新します。

次のステップ

Linux 仮想マシンに Horizon Agent をインストールします。[Linux 仮想マシンへの Horizon Agent のインストール](#)を参照してください。

構成した Linux 仮想マシンを含むデスクトップ プールを作成します。[Linux 版手動デスクトップ プールの作成](#)を参照してください。

NVIDIA ディスプレイ ドライバがインストールされているかどうかの確認

View デスクトップ セッションに NVIDIA ドライバの出力を表示して、NVIDIA ディスプレイ ドライバが RHEL 6.6/6.7/6.8 仮想マシンにインストールされていることを確認できます。

前提条件

- NVIDIA ディスプレイ ドライバをインストールしていることを確認します。
- Horizon Agent が Linux 仮想マシンにインストールされていることを確認します。[Linux 仮想マシンへの Horizon Agent のインストール](#)を参照してください。
- Linux 仮想マシンがデスクトップ プールにデプロイされていることを確認します。[Linux 版手動デスクトップ プールの作成](#)を参照してください。

手順

- 1 Linux 仮想マシンを再起動します。

Horizon Agent 起動スクリプトは、X サーバを初期化し、トポロジを表示します。

vSphere コンソールで、仮想マシンの表示を参照することはできなくなります。

- 2 Horizon Client で、Linux デスクトップに接続します。
- 3 Linux デスクトップ セッションで、NVIDIA ディスプレイ ドライバがインストールされていることを確認します。

ターミナル ウィンドウを開き、`glxinfo | grep NVIDIA` コマンドを実行します。

NVIDIA ドライバ出力が表示されます。例：

```
[root]# glxinfo | grep NVIDIA
server glx vendor string: NVIDIA Corporation
client glx vendor string: NVIDIA Corporation
OpenGL vendor string: NVIDIA Corporation
OpenGL version string: 4.5.0 NVIDIA 346.47
OpenGL shading language version string: 4.50 NVIDIA
```

ユーザーは、リモート デスクトップで NVIDIA グラフィックスの機能にアクセスできます。

NVIDIA ディスプレイ ドライバのインストールを確認した後、インストールが正しく動作するために、次のタスクを実行します。

- Linux カーネルをアップグレードする場合、Horizon Agent が View 接続サーバと通信できないことがあります。この問題を解決するには、NVIDIA ドライバを再インストールします。
- Linux 仮想マシンで NVIDIA GRID のライセンスを設定します。詳細については、NVIDIA のドキュメントを参照してください。ライセンスが設定されていない場合は、Linux デスクトップが正しく動作しません。たとえば、自動的に合わせる機能が動作しません。

vSGA を使用するための RHEL 7.2 の構成

Horizon 7 for Linux デスクトップが vSGA 機能を利用できるように、RHEL 7.2 ゲスト OS をセットアップできます。

vSGA 用の NVIDIA グラフィック カードの VIB の ESXi ホストへのインストール

ESXi 6.0 U1 以降のホストに NVIDIA GRID グラフィック カードの VIB をダウンロードしてインストールする必要があります。

NVIDIA は vSGA 用の VMware vSphere ESXi ドライバを提供しています。vSGA の場合、NVIDIA ディスプレイ ドライバは、Linux 仮想マシンにインストールされません。

前提条件

- vSphere 6.0 U1 以降のリリースが環境にインストールされていることを確認します。
- NVIDIA ドライバが環境にインストールされていることを確認します。
- NVIDIA GRID K1 または K2 GPU が ESXi ホストにインストールされていることを確認します。

手順

- 1 [NVIDIA のドライバ ダウンロード](#) サイトから NVIDIA GRID vGPU グラフィック カードの VIB をダウンロードします。

適切な VIB バージョンをドロップダウン メニューから選択します。

オプション	説明
製品タイプ	[GRID]
製品シリーズ	[GRID シリーズ] を選択します。

オプション	説明
製品	ESXi ホストにインストールされるバージョン ([GRID K2] など) を選択します。
オペレーティング システム	VMware vSphere ESXi のバージョンを選択します。

- 2 vSGA 用の VMware vSphere ESXi ドライバを ESXi 6.0 U1 ホストにアップロードします。
- 3 ESXi ホスト上のすべての仮想マシンをパワーオフまたはサスペンドします。
- 4 SSH を使用して ESXi ホストに接続します。
- 5 xorg サービスを停止します。

```
# /etc/init.d/xorg stop
```

- 6 NVIDIA VIB をインストールします。

例：

```
# esxcli system maintenanceMode set --enable true
# esxcli software vib install -v /path-to-vib/NVIDIA-VIB-name.vib
# esxcli system maintenanceMode set --enable false
```

- 7 GPU の割り当てに使用される xorg を再起動します。

```
# /etc/init.d/xorg start
```

- 8 ESXi ホストを再起動します。
- 9 ホストを再起動した後に、xorg サービスが実行されていることを確認します。

Linux 仮想マシンでの vSGA の 3D 機能の構成

RHEL 7.2 仮想マシンで vSGA を使用するように構成するには、vSphere Web Client において仮想マシンのビデオカードで 3D 設定を構成する必要があります。

前提条件

- Linux 仮想マシンをデスクトップとして使用する準備ができており、Horizon Agent がインストールされ、マシンがデスクトップ プールにデプロイされていることを確認します。
- NVIDIA VIB が ESXi ホストにインストールされていることを確認します。[vSGA 用の NVIDIA グラフィック カードの VIB の ESXi ホストへのインストール](#)を参照してください。

手順

- 1 仮想マシンをパワーオフします。
- 2 vSphere Web Client で、[仮想マシン ハードウェア] タブで仮想マシンを選択して、[設定編集] をクリックします。
- 3 [仮想ハードウェア] タブで、[ビデオ カード] をクリックして、メニュー設定を展開します。
- 4 [ビデオ メモリの合計] を 128 MB に設定します。

- 5 [3D グラフィックス] には、[3D サポートを有効化] を選択します。
- 6 [3D レンダラ] には、ドロップダウン メニューから [ハードウェア] を選択します。
- 7 [3D メモリ] には、アプリケーションの要件に合った値を選択します。

ユーザーが 3 台以上のモニタに接続する場合、この値は少なくとも 1024 MB に設定します。

- 8 [OK] をクリックします。
- 9 仮想マシンをパワーオンします。

次のステップ

vSGA が Linux 仮想マシンで実行されていることを確認します。

次に、Linux 仮想マシンに Horizon Agent をインストールします。[Linux 仮想マシンへの Horizon Agent のインストール](#)を参照してください。

vSGA が Linux 仮想マシンで実行されているかどうかの確認

仮想マシンのログ ファイルとゲスト OS を確認して、RHEL 7.2 仮想マシンで vSGA が実行されていることを確認できます。

手順

- 1 仮想マシンの `vmware.log` ファイルを開きます。

サポート対象の GPU および NVIDIA VIB が正しくインストールされている場合、ログ ファイルに次の例のような行が表示されます。

```
2015-06-24T22:19:25.259Z| mks| I120: OpenGL Version: "4.0.0 NVIDIA 346.69" (4.0.0)
2015-06-24T22:19:25.259Z| mks| I120: GLSL Version: "4.00 NVIDIA" (4.00.0)
2015-06-24T22:19:25.259Z| mks| I120: OpenGL Vendor: "NVIDIA Corporation"
2015-06-24T22:19:25.259Z| mks| I120: OpenGL Renderer: "Quadro 4000/PCIe/SSE2"
```

サポート対象の GPU および NVIDIA VIB が正しくインストールされていない場合、仮想マシンはソフトウェアレンダラーを使用します。`vmware.log` ファイルには、次の例のような行が表示されます。

```
2015-07-06T17:09:26.423Z| vmx| I120: [msg.mks.noGPUResourceFallback] Hardware GPU resources are
not available. The virtual machine uses software rendering.
2015-07-06T17:09:26.423Z| vmx| I120: -----
2015-07-06T17:09:26.425Z| svga| I120: MKS-SWP: plugin started - llvmpipe (LLVM 3.3, 256 bits)
2015-07-06T17:09:26.426Z| svga| I120: Started Shim3D
2015-07-06T17:09:26.426Z| svga| I120: MKS-RenderMain: Starting SWRenderer
```

- 2 仮想マシンのゲスト OS で、次のコマンドを入力します。

```
glxinfo|grep Gallium
```

vSGA が動作している場合、コマンドは次のテキストを返します。

```
OpenGL renderer string: Gallium 0.4 on SVGA3D; build : RELEASE;
```

vSGA が正しく動作していない場合、コマンドは次のテキストを返します。

```
OpenGL renderer string: Gallium 0.4 on llvmpipe (LLVM 3.3, 256 bits)
```

次のステップ

Linux 仮想マシンに Horizon Agent をインストールします。[Linux 仮想マシンへの Horizon Agent のインストール](#)を参照してください。

Horizon Agent のインストール

View 接続サーバがデスクトップと通信し管理することができるように、Horizon Agent を Linux デスクトップにインストールする必要があります。

この章には、次のトピックが含まれています。

- [Linux 仮想マシンへの Horizon Agent のインストール](#)
- [Linux Agent 用証明書の構成](#)
- [Linux 仮想マシンでの Horizon Agent のアップグレード](#)
- [Horizon 7 for Linux マシンをアンインストール](#)

Linux 仮想マシンへの Horizon Agent のインストール

Linux 仮想マシンをリモート デスクトップとして展開できるようにするには、Horizon Agent を Linux 仮想マシンにインストールする必要があります。

Horizon 7.0.1 のリリースで、Linux 版 Horizon Agent は、vCenter Server の管理対象仮想マシンを使用します。管理対象仮想マシンは、次の機能強化を提供します。

- vCenter Server は、Linux デスクトップ デプロイ環境では必須要件です。
- Linux での Horizon Agent のインストールには、登録は必要ありません。
- ほとんどの Linux デスクトップ デプロイ環境では、ベース仮想マシンに Horizon Agent をインストールできます。

注意: NVIDIA GRID vGPU、vDGA、または vSGA を使用する場合には、Horizon Agent をインストールする前に、Linux 仮想マシンでこれらの 3D 機能を構成する必要があります。Horizon Agent を最初にインストールしてしまうと、`xorg.conf` ファイルの必須パラメータが上書きされ、3D グラフィックス機能が動作しません。

[vGPU を使用するための RHEL 6.6/6.7/6.8 と RHEL 7.2 の構成](#)、[vDGA を使用するための RHEL 6.6/6.7/6.8 の構成](#)、または [vSGA を使用するための RHEL 7.2 の構成](#)を参照してください。3D グラフィックスの構成が完了したら、Horizon Agent をインストールします。

2D グラフィックスを構成する場合は、[リモート デスクトップ デプロイ用の Linux マシンの準備](#)の手順を完了した後、Horizon Agent をインストールできます。

前提条件

- Linux ゲスト OS がデスクトップとして使用できるように準備されていることを確認します。 [リモート デスクトップ デプロイ用の Linux マシンの準備](#)を参照してください。
- Linux 用の Horizon Agent インストーラ スクリプトについて理解しておきます。 [install_viewagent.sh コマンドライン オプション](#)を参照してください。

手順

- 1 VMware ダウンロード サイト (<https://my.vmware.com/web/vmware/downloads>) から、Linux 版 Horizon Agent インストーラ ファイルをダウンロードします。

[Desktop & End-User Computing] にある VMware Horizon 7 のダウンロードを選択します。このダウンロードには、Linux 版 Horizon Agent のインストーラが含まれます。

y.y.y がバージョン番号で、*xxxxxxx* がビルド番号の場合、インストーラ ファイル名は、64 ビット Linux で VMware-horizonagent-linux-x86_64-*y.y.y*-*xxxxxxx*.tar.gz です。

- 2 お使いの Linux ディストリビューションの tar ボールをゲスト OS に展開します。

例：

```
tar -xzf <View Agent の tar ボール>
```

- 3 tar ボール フォルダに移動します。
- 4 `install_viewagent.sh` スクリプトをスーパーユーザーとして実行します。

コマンドライン オプションのリストについては、 [install_viewagent.sh コマンドライン オプション](#)を参照してください。

例：

```
sudo ./install_viewagent.sh
```

- 5 `-A` オプションを指定せずに `install_viewagent.sh` を実行する場合、**Yes** と入力して EULA に同意します。
EULA に同意しない限りインストーラは実行されません。
- 6 Linux を再起動して変更を有効にします。

インストール後に、*viewagent* サービスは開始されます。サービスが、`sudo service viewagent status` を使用して開始されたことを確認します。

次のステップ

デスクトップ プールに仮想マシンをデプロイします。 [Linux 版手動デスクトップ プールの作成](#)を参照してください。

install_viewagent.sh コマンドライン オプション

`install_viewagent.sh` スクリプトは、Horizon Agent を Linux ゲスト OS にインストールします。

gnome デスクトップ環境のコマンド ウィンドウで、次の形式の `install_viewagent.sh` スクリプトを使用します。

```
install_viewagent.sh command_option argument [command_option argument] . . .
```

`install_viewagent.sh` スクリプトには、必須およびオプションのパラメータが含まれます。

表 5-1. `install_viewagent.sh` 必須のオプション パラメータ

オプション パラメータ (必須情報)	説明
-A yes no	EULA を受け入れまたは拒否します。インストールを続行するには、 yes を指定する必要があります。

表 5-2. `install_viewagent.sh` のオプション パラメータ

オプション パラメータ	説明
-a yes no	オーディオ入力サポートをインストールするかバイパスします。デフォルトは、 no です。
-s	自己署名証明書 subject DN。デフォルトでは、インストーラは、Blast を使用します。
-m yes no	スマートカード リダイレクト サポートをインストールするかバイパスします。デフォルトは、 no です。
-j	JMS SSL キーストア パスワード。デフォルトでは、インストーラは任意の文列を生成します。
-r yes no	インストール後にシステムを自動的に再起動します。デフォルトは、 いいえ です。
-M yes no	Linux エージェントを、管理された、または管理されていないエージェントにアップグレードします。デフォルトは、 yes です。

表 5-3. install_viewagent.sh パラメータの例

状況	例
新規インストール	<pre>sudo ./install_viewagent.sh -A yes</pre> <p>新規インストールでは、常に新しいデスクトップ プールの作成が必要です。</p>
非管理対象仮想マシンからアップグレードし、管理対象仮想マシンのスタイルを保持します。	<pre>sudo ./install_viewagent.sh -A yes-M no</pre> <p>このタイプのアップグレードには、新しいデスクトップ プールの作成は必要ありません。既存のデスクトップ プールを再使用できます。</p>
非管理対象仮想マシンのデプロイからアップグレードし、管理対象仮想マシンのスタイルへ変換します。アップグレードには、ブローカーに新しいデスクトップ プールの作成が必須です。	<pre>sudo ./install_viewagent.sh -A yes</pre> <p>このタイプのアップグレードには、新しいデスクトップ プールの作成が必要です。既存のデスクトップ プールを削除する必要があります。</p>

Linux Agent 用証明書の構成

Linux Agent をインストールすると、インストーラによって VMwareBlastServer の自己署名証明書が生成されます。

- Blast Security Gateway がブローカーで無効になっているとき、VMwareBlastServer は、この証明書を HTML Access を使用するブラウザに提示して Linux デスクトップに接続します。
- Blast Security Gateway がブローカーで有効になっているとき、Blast Security Gateway の証明書は、この証明書をブラウザに提示します。

業界またはセキュリティの規制に準拠するために、自己署名証明書に代わって認証局 (CA) が署名した証明書を使用できます。

手順

- 1 VMwareBlastServer のプライベート キーと証明書をインストールします。
 - a プライベート キーを rui.key に名前変更し、証明書を rui.crt に名前変更します。
 - b `sudo chmod 550 /etc/vmware/ssl` を実行します。
 - c rui.crt と rui.key を /etc/vmware/ssl にコピーします。
 - d `chmod 440 /etc/vmware/ssl` を実行します。

2 ルート認証局と中間認証局を Linux OS の認証局ストアにインストールします。

注: Linux システムの設定変更については、お使いの Linux ディストリビューションのドキュメントを確認してください。

Linux 仮想マシンでの Horizon Agent のアップグレード

Horizon Agent の最新バージョンをインストールすることにより、Linux マシンで Horizon Agent をアップグレードできます。

非管理対象仮想マシン：エージェントインストーラは、ブローカ Admin 情報を必要とするブローカにマシンを登録します。デスクトップ プール作成ウィザードは、マシン ソース ページの [その他のソース] を使用して、登録されたマシンを選択します。

管理対象仮想マシン：インストーラはブローカと通信を行いません。デスクトップ プール作成ウィザードは、マシン ソース ページの [vCenter Server 仮想マシン] を使用して、vCenter Server の仮想マシンを選択します。管理対象仮想マシン展開は、以下の機能をサポートします。

- リモート マシンの電源ポリシー
- ユーザーによるマシンのリセットを許可

注: Horizon Agent for Linux 7.0.0 を含む以前のバージョンは、非管理対象仮想マシンとして機能していました。Horizon Agent for Linux 7.0.1 は、管理対象仮想マシンサポートとして機能します。

非管理対象から管理対象仮想マシン展開にアップグレードする場合、以下の方法を使用できます。

- 非管理対象仮想マシン展開を保持し、必要なバージョンにアップグレードします。このタイプのアップグレードでは、View 接続サーバでの構成変更は必要ありません。
- 非管理対象仮想マシン展開から任意のバージョンの管理対象仮想マシン展開にアップグレードします。このタイプのアップグレードでは、View 接続サーバでの新しいデスクトップ プールの作成が必要になります。

注: 管理対象仮想マシン展開からのアップグレードの場合、管理対象仮想マシン展開を保持し、必要なバージョンにアップグレードできます。ただし、アップグレード時に管理対象仮想マシン展開から非管理対象仮想マシン展開への変換はサポートされていません。

アップグレードでは以下のパラメータを利用できます。

パラメータ	説明
-A yes	EULA への同意。インストールを続行するには、 yes を指定する必要があります。このパラメータを指定しないと、インストールスクリプトで値が要求されます。
-m yes	スマート カード リダイレクトを有効にします。
-r	インストール後にオペレーティング システムを再起動します。使用可能な値は、 yes および no です。デフォルトは、 no です。
-M yes no	Linux Agent を管理対象 非管理対象エージェントにアップグレードします。デフォルト値は yes です。

Linux 仮想マシンでの Horizon Agent のアップグレード

Horizon Agent の最新バージョンをインストールすることにより、Linux マシンで Horizon Agent をアップグレードできます。

前提条件

- VMwareBlastServer プロセスが実行されていないことを確認します。

このプロセスを停止するには、ユーザーがマシンからログオフしていて、アクティブなデスクトップ セッションがないことを確認するか、マシンを再起動します。

手順

- 1 VMware ダウンロード サイト (<https://my.vmware.com/web/vmware/downloads>) から、最新の Linux 版 Horizon Agent インストーラ ファイルをダウンロードします。

[Desktop & End-User Computing] にある VMware Horizon 7 のダウンロードを選択します。このダウンロードには、Linux 版 Horizon Agent のインストーラが含まれます。

インストーラ ファイル名は、64 ビット Linux で VMware-viewagent-linux-x86_64-y.y.y — xxxxxx.tar.gz です。y.y.y はバージョン番号、xxxxxx はビルド番号です。

- 2 お使いの Linux ディストリビューションの tar ボールをゲスト OS に展開します。

例：

```
tar -xzf <View Agent の tar ボール>
```

- 3 tar ボール フォルダに移動します。
- 4 install_viewagent.sh スクリプトを実行し、以下の展開シナリオの 1 つを使用して非管理対象仮想マシンをアップグレードします。

オプション	説明
非管理対象仮想マシン展開をアップグレードし、非管理対象仮想マシン展開を保持する	<code>sudo ./install_viewagent.sh -A yes -M no</code>
非管理対象仮想マシン展開をアップグレードし、管理対象仮想マシン展開に変更する	<code>sudo ./install_viewagent.sh -A yes -M yes</code> 注： View Administrator で、非管理対象仮想マシン展開用の既存のデスクトップ プールを削除し、管理対象仮想マシン展開用の新しいデスクトップ プールを作成します。詳細については、 Linux 版手動デスクトップ プールの作成 を参照してください。
管理対象仮想マシン展開をアップグレードする	<code>sudo ./install_viewagent.sh -A yes -M yes</code> 注： アップグレード後、既存のデスクトップ プールを再利用できます。

Horizon 7 for Linux マシンをアンインストール

仮想マシンで Horizon 7 for Linux をアンインストールするには、Horizon Agent をアンインストールし、構成ファイルを削除する必要があります。

前提条件

VMwareBlastServer プロセスが実行されていないことを確認します。このプロセスを停止するには、マシンからログオフしていて、アクティブなデスクトップ セッションがないことを確認するか、マシンを再起動します。

手順

- 1 仮想マシンでターミナル ウィンドウを開き、Horizon Agent のアンインストール スクリプトを実行します。

```
sudo /usr/lib/vmware/viewagent/bin/uninstall_viewagent.sh
```

スクリプトは、Horizon Agent のプロセスを停止し、インストール ディレクトリ `/usr/lib/vmware/viewagent` から Horizon Agent サービスとソフトウェアを削除します。

- 2 `/etc/vmware` のディレクトリから、Horizon 7 for Linux の構成ファイルを手動で削除します。

Linux デスクトップの構成オプション

構成ファイルを使用してさまざまなオプションを構成し、ユーザーの使用環境をカスタマイズできます。

この章には、次のトピックが含まれています。

- Linux デスクトップでの構成ファイルのオプション設定
- Linux デスクトップの Blast 設定の例
- Linux デスクトップの vSphere コンソールへの表示を抑制する

Linux デスクトップでの構成ファイルのオプション設定

/etc/vmware/config ファイルまたは /etc/vmware/viewagent-custom.conf ファイルにエントリを追加して、特定のオプションを構成できます。

インストーラは、View Agent または Horizon Agent のインストール中に、2 つの構成テンプレート ファイル config.template と viewagent-custom.conf.template を /etc/vmware にコピーします。/etc/vmware/config ファイルと /etc/vmware/viewagent-custom.conf ファイルが存在しない場合、インストーラは config.template を config に、viewagent-custom.conf.template を viewagent-custom.conf にコピーします。テンプレート ファイルではすべての構成オプションがリストされていて、詳細な説明があります。オプションを設定するには、コメントを削除して値を適切に変更します。

たとえば、/etc/vmware/config の次の行により、可逆圧縮 PNG モードが有効になります。

```
RemoteDisplay.alwaysLossless=TRUE
```

構成を変更したら、Linux を再起動して変更を有効にしてください。

/etc/vmware/config の構成オプション

VMwareBlastServer およびその関連プラグインでは、構成ファイル /etc/vmware/config が使用されます。

表 6-1. /etc/vmware/config の構成オプション

オプション	値	デフォルト	説明
RemoteDisplay.alwaysLossless	TRUE または FALSE	FALSE	特にグラフィック設計アプリケーションなどのグラフィック アプリケーションでは、Linux デスクトップのクライアント表示で正確なピクセル レベルの画像処理が必要となります。Linux デスクトップで生成されクライアント デバイスで処理される画像とビデオ再生については、可逆圧縮 PNG モードを構成できます。この機能では、クライアントと ESXi ホストの間で追加の帯域幅が使用されます。
mksVNCServer.useUInputButtonMapping	TRUE または FALSE	FALSE	Ubuntu または RHEL 7 で左手用マウスのサポートを有効にします。CentOS と RHEL 6.6/6.7/6.8 では左手用マウスがサポートされており、このオプションを設定する必要はありません。
RemoteDisplay.allowAudio	TRUE または FALSE	TRUE	オーディオ出力を無効にします。
WC.ScRedir.Enable	TRUE または FALSE	TRUE	スマート カード リダイレクトを無効にします。
WC.logLevel	FATAL、ERROR、WARN、INFO、DEBUG、または TRACE	INFO	このオプションを使用して、WC プロキシ ノードのログ レベルを設定します。
WC.RTAV.Enable	TRUE または FALSE	TRUE	このオプションを設定してオーディオ入力を無効にします。
Clipboard.Direction	0、1、2、3	2	このオプションにより、クリップボード リダイレクト ポリシーが決定されます。 <ul style="list-style-type: none"> ■ 0 - クリップボード リダイレクトを無効にします。 ■ 1 - クリップボード リダイレクトを両方向で有効にします。 ■ 2 - クリップボード リダイレクトをクライアントからリモート デスクトップのみで有効にします。 ■ 3 - クリップボード リダイレクトをリモート デスクトップからクライアントのみで有効にします。
mksVNCServer.useXExtButtonMapping	TRUE または FALSE	FALSE	このオプションを設定して SLED 11 SP3 での左手用マウスのサポートを有効または無効にします。
mksvhan.clipboardSize	INTEGER	1024	このオプションを使用して、クリップボードの最大サイズをコピーおよび貼り付けます。
RemoteDisplay.maxBandwidthKbps	INTEGER	409600	VMware Blast セッションの最大帯域幅をキロビット/秒 (kbps) 単位で指定します。この帯域幅には、イメージ、オーディオ、仮想チャネル、および VMware Blast 制御のすべてのトラフィックが含まれます。最大値は、4 Gbps (4096000) です。
RemoteDisplay.maxFPS	INTEGER	60	画面更新の最大レートを指定します。この設定を使用して、ユーザーが使用する平均帯域幅を管理します。有効値は 3 から 60 までの間にする必要があります。デフォルトは 1 秒あたり 60 回の更新です。
RemoteDisplay.enableStats	TRUE または FALSE	FALSE	帯域幅、FPS、RTT などでは、Blast protocol statistics を mks ログで有効または無効にします。

オプション	値	デフォルト	説明
RemoteDisplay.allowH264	TRUE または FALSE	TRUE	H.264 エンコードを有効または無効にするようにこのオプションを設定します。
vdpservice.log.logLevel	FATAL、ERROR、WARN、INFO、DEBUG、または TRACE	INFO	このオプションを使用して、vdpservice のログレベルを設定します。
RemoteDisplay.qpmaxH264	利用可能な値の範囲：0 ~ 51	36	このオプションを使用して、H264minQP 量子化パラメータを設定します。このパラメータは、H.264 エンコードを使用するように構成されたりリモート ディスプレイの最高イメージ品質を指定します。RemoteDisplay.qpminH264 に設定した値よりも大きな値を設定します。
RemoteDisplay.qpminH264	利用可能な値の範囲：0 ~ 51	10	このオプションを使用して、H264maxQP 量子化パラメータを設定します。このパラメータは、H.264 エンコードを使用するように構成されたりリモート ディスプレイの最低イメージ品質を指定します。RemoteDisplay.qpmaxH264 に設定した値よりも小さな値を設定します。
RemoteDisplay.minQualityJPEG	利用可能な値の範囲：1 ~ 100	25	JPEG/PNG エンコードを使用する場合のデスクトップ ディスプレイのイメージ品質を指定します。低品質設定は、スクロール発生時など、頻繁に変化する画面の領域に適しています。
RemoteDisplay.midQualityJPEG	利用可能な値の範囲：1 ~ 100	35	JPEG/PNG エンコードを使用する場合のデスクトップ ディスプレイのイメージ品質を指定します。デスクトップ ディスプレイの中程度の品質を設定するために使用します。
RemoteDisplay.maxQualityJPEG	利用可能な値の範囲：1 ~ 100	90	JPEG/PNG エンコードを使用する場合のデスクトップ ディスプレイのイメージ品質を指定します。高品質設定は、より静的な画面の領域に適していて、イメージ品質がより高くなります。

/etc/vmware/viewagent-custom.conf の構成オプション

Java Standalone Agent では、構成ファイル `/etc/vmware/viewagent-custom.conf` が使用されます。

表 6-2. /etc/vmware/viewagent-custom.conf の構成オプション

オプション	値	デフォルト	説明
サブネット	NULL、または IP アドレス/CIDR 形式のネットワークアドレスとマスク	NULL	<p>異なるサブネットを持つ複数のローカル IP アドレスがある場合、このオプションを使用して、Linux エージェントが View 接続サーバに提供するサブネットを設定します。</p> <p>Linux エージェント マシンで複数のサブネット構成が検出される場合、Linux エージェントが使用する適切なサブネットを指定するために、このオプションが必要です。たとえば、Docker を Linux マシンにインストールした場合、Docker は仮想ネットワーク アダプタとして導入されます。Linux エージェントが仮想ネットワーク アダプタとして Docker を使用しないようにするには、このオプションを設定し、実際の物理ネットワーク アダプタを使用する必要があります。</p> <p>IP アドレス/CIDR 形式で値を指定する必要があります。例： Subnet=192.168.1.0/24</p> <p>NULL は、Linux エージェントがランダムに IP アドレスを選択することを意味します。</p>
SSOEnable	TRUE または FALSE	TRUE	シングル サインオン (SSO) を無効にします。
SSOUserFormat	テキスト文字列	[username]	<p>シングル サインオンのログイン名の形式を指定します。デフォルトはユーザー名のみです。ドメイン名も要求する場合は、このオプションを設定します。一般的にログイン名では、ドメイン名と特殊文字にユーザー名を続けます。特殊文字をバックスラッシュにする場合は、別のバックスラッシュを使用してエスケープする必要があります。ログイン名の形式の例を次に挙げます。</p> <ul style="list-style-type: none"> ■ SSOUserFormat=[domain]\\[username] ■ SSOUserFormat=[domain]+[username] ■ SSOUserFormat=[username]@[domain]
StartBlastServerTimeout	整数	20	VMwareBlastServer プロセスで初期化に使用する時間を秒単位で決めます。このタイムアウト値以内にプロセスの準備ができない場合、ユーザーのログインは失敗します。
SSLCiphers	テキスト文字列	!aNULL:kECDH+AESGCM:ECDH+AESGCM:RSA+AESGCM:kECDH+AES:ECDH+AES:RSA+AES	暗号化のリストを指定します。 https://www.openssl.org/docs/manmaster/apps/ciphers.html で定義されている形式を使用する必要があります。
SSLProtocols	テキスト文字列	TLSv1_1:TLSv1_2	セキュリティ プロトコルを指定します。サポートされるプロトコルは、TLSv1.0、TLSv1.1、TLSv1.2 です。
SSLCipherServerPreference	TRUE または FALSE	TRUE	オプション SSL_OP_CIPHER_SERVER_PREFERENCE を有効または無効にします。詳細については、 https://www.openssl.org/docs/manmaster/ssl/SSL_CTX_set_options.html を参照してください。
LogCnt	整数	-1	<p>このオプションを使用して、/tmp/vmware-root に保持するログ ファイルの数を設定します。</p> <ul style="list-style-type: none"> ■ -1 - すべて保持 ■ 0 - すべて削除 ■ > 0 - 保持するログの数。

オプション	値	デフォルト	説明
RunOnceScript			<p>このオプションを使用して、クローン作成された仮想マシンを Active Directory へ再参加させます。</p> <p>ホスト名の変更後、RunOnceScript を設定します。指定されたスクリプトは、最初のホスト名の変更後、一度だけ実行されます。Agent サービスが開始され、ホスト名が Agent のインストール後に変更された場合、スクリプトは root 権限として実行されます。</p> <p>たとえば、winbind ソリューションでは、winbind でベース仮想マシンを Active Directory に参加させ、このオプションをスクリプトパスに設定する必要があります。これには、ドメインへ再度参加させるコマンド <code>/usr/bin/net ads join -U <ADUserName> %<ADUserPassword></code> が含まれている必要があります。仮想マシンのクローン作成後、オペレーティング システムのカスタマイズによってホスト名が変更されます。Agent サービスが開始されると、クローン作成された仮想マシンを Active Directory へ参加するスクリプトが実行されます。</p>
RunOnceScriptTimeout		120	<p>このオプションを使用して、RunOnceScript オプションのタイムアウト値を秒数で設定します。</p> <p>たとえば、<code>RunOnceScriptTimeout=120</code> のように設定します。</p>

注: 3つのセキュリティ オプション、SSLCiphers、SSLProtocols、SSLCipherServerPreference は VMwareBlastServer プロセス用です。VMwareBlastServer プロセスが開始されると、Java Standalone Agent はこれらのオプションをパラメータとして渡します。Blast Secure Gateway (BSG) が有効であるとき、これらのオプションは BSG と Linux デスクトップの間の接続に影響します。BSG が無効であるとき、これらのオプションはクライアントと Linux デスクトップの間の接続に影響します。

Linux デスクトップの Blast 設定の例

リモート デスクトップ ディスプレイのイメージ品質を調整して、ユーザーの使用環境を向上できます。イメージ品質を向上すると、ネットワーク接続が不安定になる場合でも、一貫性のあるユーザーの使用環境を維持できます。

VMware Blast Extreme プロトコル設定の例

VMwareBlastServer およびその関連プラグインでは、構成ファイル `/etc/vmware/config` が使用されます。

表 6-3. `/etc/vmware/config` の Blast 構成オプションの例

オプション名	パラメータ	高速 LAN	LAN	専用 WAN	ブロードバンド WAN	低速 WAN	超低速
帯域幅設定	RemoteDisplay.maxBandwidthKbps	1000000 (1 Gbps)	1000000 (1 Gbps)	1000000 (1 Gbps)	5000 (5 Mbps)	2000 (2 Mbps)	1000 (1 Mbps)
最大 FPS	RemoteDisplay.maxFPS	60	30	30	20	15	5
オーディオ再生	RemoteDisplay.allowAudio	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
表示品質 (JPEG/PNG)	RemoteDisplay.maxQualityJPEG	90	90	90	70	60	50

オプション名	パラメータ	高速 LAN	LAN	専用 WAN	ブロードバンド WAN	低速 WAN	超低速
表示品質 (JPEG/PNG)	RemoteDisplay.midQualityJPEG	35	35	35	35	35	35
表示品質 (JPEG/PNG)	RemoteDisplay.minQualityJPEG	25	25	25	20	20	20
表示品質 (H.264)	RemoteDisplay.qpmaxH264	28	36	36	36	36	42
表示品質 (H.264)	RemoteDisplay.qpminH264	10	10	10	10	10	10

Linux デスクトップの vSphere コンソールへの表示を抑制する

ユーザーが Linux デスクトップに接続するときに、このデスクトップを Linux 仮想マシンの vSphere コンソールに表示させることもできます。ユーザーがデスクトップに接続するときには vSphere コンソールを必ず空白にするように Linux 仮想マシンを構成できます。

手順

- ◆ ESXi ホストで、Linux 仮想マシンの vmx ファイルに次の行を追加します。

```
RemoteDisplay.maxConnections = "0"
```

ユーザーがデスクトップからログアウトしたときに仮想マシンに接続するときでも、vSphere コンソールの表示は空白のままになります。

Linux デスクトップ プールの作成と管理

7

Linux 仮想マシンを構成してリモート デスクトップとして使用するには、Linux 仮想マシンのデスクトップ プールを作成する必要があります。

Horizon for Linux では、次のデスクトップ プール タイプがサポートされます。

- vCenter Server の仮想マシンがある手動デスクトップ プール
- 自動化される完全なクローン デスクトップ プール

vCenter Server の仮想マシンがある手動デスクトップ プールを作成するには、すべての仮想マシンに Horizon Agent をインストールする必要があります。次に、接続サーバ デスクトップ プール作成ウィザードを使用して、仮想マシンをデスクトップ プールに追加します。多くの仮想マシンのクローンを作成する方法の詳細は、[Linux デスクトップの一括デプロイの概要](#)を参照してください。

自動化された完全なクローン デスクトップ プールを作成するには、Linux 仮想マシン テンプレートに Horizon Agent をインストールする必要があります。次に、接続サーバ デスクトップ プール作成ウィザードを使用して、完全な仮想マシンのクローンを作成します。

この章には、次のトピックが含まれています。

- [Linux 版手動デスクトップ プールの作成](#)
- [Linux デスクトップ プールの管理](#)
- [Linux の自動化された完全なクローン デスクトップ プールの作成](#)
- [ブローカ PowerCLI コマンド](#)

Linux 版手動デスクトップ プールの作成

Linux 仮想マシンの手動デスクトップ プールを作成できます。

前提条件

- Horizon Agent が Linux ゲスト OS にインストールされていることを確認します。[Linux 仮想マシンへの Horizon Agent のインストール](#)を参照してください。
- vCenter Server が Horizon 接続サーバに追加されていることを確認します。

手順

- 1 View Administrator で、手動デスクトップ プールを追加します。
[カタログ] - [デスクトップ プール] - [追加] を選択します。
- 2 [手動デスクトップ プール] を選択します。
- 3 [ユーザー割り当て] ページで、デスクトップ プールのマシンに専用ユーザーまたはフローティング ユーザー割り当てのどちらかを選択して、[次へ] をクリックします。
- 4 [マシン ソース] ページで、[vCenter 仮想マシン] を選択して、[次へ] をクリックします。
- 5 [vCenter Server] ページで、適切な vCenter Server を選択して、[次へ] をクリックします。
- 6 **デスクトップ プール ID** を入力します。
- 7 [デスクトップ プールの設定] ページで、次のオプションを設定します。

オプション	説明
デフォルト表示プロトコル	VMware Blast
ユーザーがプロトコルを選択できるようにする	いいえ
3D レンダラー	2D 版 vSphere Client、vSGA、または、vDGA デスクトップ、および vGPU デスクトップ版 NVIDIA GRID vGPU を使用して管理します。

注: プールの設定は必須です。設定していない場合、デスクトップへの接続が失敗して、プロトコル エラーやブラック スクリーンが生じる場合があります。

- 8 [vCenter 仮想マシンを追加] ページで、[Linux 仮想マシン] を選択します。[追加] をクリックし、[次へ] をクリックします。

注: 同じデスクトップ プールに、Windows と Linux の仮想マシンを作成しないでください。

- 9 プロンプトに従って、手順を完了します。

デスクトップ プールにあるマシンへの資格をユーザーに付与します。View Administrator で、[デスクトップ プール] を選択し、[資格] > [資格を追加] を選択して、ユーザーまたはグループを追加します。

Linux 仮想マシンを、Horizon 7 を展開した環境でリモート デスクトップとして使用する準備が整いました。

Linux デスクトップ プールの管理

手動デスクトップ プールを作成し、プールに Linux マシンを追加すると、設定構成により手動デスクトップ プールを管理できます。手動デスクトップ プールには、Linux ゲスト オペレーティング システムだけを追加する必要があります。プールに Windows と Linux ゲスト オペレーティング システムの両方が含まれる場合、プールは Windows プールとして扱われ、Linux デスクトップに接続できなくなります。

操作管理のサポート

- デスクトップ プールの無効化または有効化

- 自動デスクトップ プールのクローン作成
- デスクトップ プールの削除

View Manager から仮想マシンを取り外したり、ディスクから仮想マシンを削除できます。

リモート設定のサポート

表 7-1. リモート設定

リモート設定	オプション
リモート マシンの電源ポリシー	<ul style="list-style-type: none"> ■ 電源操作を行わない ■ マシンは常にパワーオン ■ サスペンド ■ パワーオフ
切断後に自動的にログオフ	<ul style="list-style-type: none"> ■ 直ちに実行 ■ 実行しない ■ n 分後に
ユーザーによるマシンのリセットを許可	<ul style="list-style-type: none"> ■ はい ■ いいえ
ユーザーが複数のクライアント デバイスからセッションを個別に開始できるようにする	<ul style="list-style-type: none"> ■ はい ■ いいえ
完全なクローンがある自動デスクトップ プールおよびフローティング デスクトップ プールで「ログオフ後にマシンを削除」	<ul style="list-style-type: none"> ■ はい ■ いいえ

View Administrator 操作のサポート

- セッションを切断
- Logoff Session (セッションのログオフ)
- メッセージを送信

専用デスクトップ プールについては、各仮想マシンのユーザー割り当てを追加したり削除できます。多くの操作を行うには、View PowerCLI Cmdlets を使用する必要があります。

- Update-UserOwnership

■ Remove-UserOwnership

注: [リモート表示プロトコル] 設定は変更しないでください。この設定は、常にデスクトップ プール作成と同じ必要があります。

設定	オプション
デフォルト表示プロトコル	VMware Blast
ユーザーがプロトコルを選択できるようにする	いいえ
3D レンダラー	<ul style="list-style-type: none"> ■ 2D、vSGA、または vDGA の vSphere Client を使用して管理する ■ NVIDIA GRID vGPU

詳細については、『View 管理』ガイドを参照してください。

Linux の自動化された完全なクローン デスクトップ プールの作成

Linux 仮想マシンの自動化された完全なクローン デスクトップ プールを作成できます。自動化された完全なクローン デスクトップ プールを作成した後は、Horizon 7 のデプロイ環境で Linux 仮想マシンをリモート デスクトップとして使用できます。

前提条件

- Horizon Agent が Linux ゲスト OS にインストールされていることを確認します。[Linux 仮想マシンへの Horizon Agent のインストール](#)を参照してください。
- Winbind ソリューションを使用して、Linux 仮想マシンを Active Directory に参加させる場合、仮想マシン テンプレートで Winbind ソリューションの構成を完了する必要があります。
- Winbind ソリューションを使用する場合、仮想マシンでドメインに参加するためのコマンドを実行する必要があります。シェル スクリプトにこのコマンドを追加して、/etc/vmware/viewagent-custom.conf にある Horizon Agent のオプション RunOnceScript にこのスクリプトのパスを指定します。詳細については、[Linux デスクトップでの構成ファイルのオプション設定](#)を参照してください。
- vCenter Server が Horizon 接続サーバに追加されていることを確認します。

手順

- 1 ゲスト カスタマイズの仕様を作成します。

vSphere 仮想マシン管理ドキュメントの「vSphere Web Client での Linux 向けカスタマイズ仕様の作成」を参照してください。仕様を作成する場合、次の設定を必ず正しく指定してください。

設定	値
ターゲット仮想マシンの OS	Linux
コンピュータ名	仮想マシン名を使用します。
ドメイン	View 環境のドメインを指定します。

設定	値
ネットワーク設定	標準ネットワーク設定を使用します。
プライマリ DNS	有効なアドレスを指定します。

注: ゲスト OS のカスタマイズのサポート一覧の詳細については、<http://partnerweb.vmware.com/programs/guestOS/guest-os-customization-matrix.pdf> を参照してください。

- Horizon Administrator で、[カタログ] - [デスクトップ ツール] - [追加] を選択します。
- [自動化されたデスクトップ プール] を選択して、[次へ] をクリックします。
- デスクトップ プールにあるマシンのユーザー割り当てについて「専用」または「流動」のいずれかを選択して、[次へ] をクリックします。
- [vCenter Server] ページで、[フル仮想マシン] を選択し、適切な vCenter Server を選択して、[次へ] をクリックします。
- [デスクトップ プール ID] ページで、デスクトップ プール ID を入力して、[次へ] をクリックします。
- [デスクトップ プールの設定] ページで、次のオプションを設定して、[次へ] をクリックします。

オプション	説明
デフォルト表示プロトコル	VMware Blast
ユーザーがプロトコルを選択できるようにする	いいえ
3D レンダラー	2D 版 vSphere Client、vSGA、または、vDGA デスクトップ、および vGPU デスクトップ版 NVIDIA GRID vGPU を使用して管理します。

注: プールの設定は必須です。設定していない場合、デスクトップへの接続が失敗して、プロトコル エラーやブラック スクリーンが生じる場合があります。

- [プロビジョニングの設定] ページで、[仮想マシンの名前付け] オプションを設定して、[次へ] をクリックします。

オプション	説明
名前を手動で指定	名前を手動で入力します。
名前付けパターンを使用	たとえば、LinuxVM-{n} のように指定します。 次のデスクトップ プールのサイズ設定オプションを指定する必要があります。 ■ マシンの最大数 ■ スペアのパワーオン状態のマシンの数

- [ストレージの最適化] ページで、ストレージ管理のポリシーを選択して、[次へ] をクリックします。
- [vCenter 設定] ページで、[参照] をクリックし、vCenter Server の設定を順番に選択し、[次へ] をクリックする必要があります。

vCenter Server の設定を省略することはできません。

- テンプレート

- b 仮想マシンのフォルダの場所
- c ホストまたはクラスタ
- d リソース プール
- e データストア

- 11 [詳細ストレージ オプション] ページで、適切なストレージ オプションを選択して、[次へ] をクリックします。
- 12 [ゲストのカスタマイズ] ページで、Linux のゲスト カスタマイズを選択して、[次へ] をクリックします。
- 13 [設定内容の確認] ページで、詳細を確認して、[このウィザードの終了後にユーザーに資格を割り当てる] を選択します。
- 14 [終了] をクリックします。
- 15 デスクトップ プールにあるマシンへの資格をユーザーに付与するには、デスクトップ プールを選択して、[資格] - [資格を追加] をクリックして、ユーザーとグループを追加します。
- 16 デスクトップ プールのすべての Linux 仮想マシンが利用可能になるまで待機します。

ブローカ PowerCLI コマンド

接続サーバや Windows デスクトップでさまざまな管理タスクを実行する View PowerCLI cmdlets は、Linux デスクトップで動作できます。

手動デスクトップ プールの作成

```
Add-ManualPool -DefaultProtocol Blast -AllowProtocolOverride $false -threadRender usevc|vgpu -  
Pool_id <pool id> [more parameters]
```

Linux デスクトップでは、次のオプションは必須です。

- DefaultProtocol Blast
- AllowProtocolOverride \$false
- threadRender usevc|vgpu.vGPU デスクトップでは -threadRender vgpu を使用し、2D/vSGA/vDGA デスクトップでは -threadRender usevc を使用します。

[例]

- 仮想マシン LinuxVM-01 で、LinuxDesktop という名のフローティングの Linux デスクトップ プールを作成します。

```
Add-ManualPool -DefaultProtocol Blast -AllowProtocolOverride $false -threadRender usevc -Pool_id  
LinuxDesktop -Id (Get-DesktopVM -Name LinuxVM-01).id -Persistence NonPersistent -Vc_name  
myvc.myorg.org
```

- LinuxVM- という仮想マシン名で始まるすべての仮想マシンを持つ LinuxDesktop という名の専用 Linux vGPU デスクトップ プールを作成します。

```
Get-DesktopVM | Where-Object {$_.Name.StartsWith("LinuxVM-")} | Add-ManualPool -DefaultProtocol Blast -AllowProtocolOverride $false -Persistence Persistent -threadRender vgpu -Pool_id LinuxDesktop
```

- 最初の RHEL 6 x64 仮想マシンでフローティング Linux デスクトップ プール LinuxDesktop を作成します。

```
Get-DesktopVM | Where-Object {$_.GuestID -eq "rhel6_64Guest"} | Select-Object -Index 0 | Add-ManualPool -DefaultProtocol Blast -AllowProtocolOverride $false -Persistence NonPersistent -threadRender usevc -Pool_id LinuxDesktop
```

完全なクローンの自動化されたデスクトップ プールを作成

```
Add-AutomaticPool -DefaultProtocol Blast -AllowProtocolOverride $false -threadRender usevc|vgpu `
-Pool_id <pool id> -Vc_id <vCenter id> `
-NamePrefix <VM Name Prefix> " `
-templatePath <Virtual Machine Template Path> `
-VmFolderPath <Virtual Machine Folder Path> `
-ResourcePoolPath <Resource Pool Path> `
-dataStorePaths <Datastore Path> `
-customizationSpecName <Customization Specification Name> `
[more parameters]
```

Linux デスクトップでは、次のオプションは必須です。

- DefaultProtocol Blast
- AllowProtocolOverride \$false
- threadRender usevc|vgpu.vGPU デスクトップでは -threadRender vgpu を使用し、2D/vSGA デスクトップでは -threadRender usevc を使用します。

[例]

```
Add-AutomaticPool -DefaultProtocol Blast -AllowProtocolOverride $false -threadRender usevc `
-pool_id FullClone-Linux `
-Vc_id (Get-ViewVC -serverName myvc.myorg.org).vc_id `
-NamePrefix "FullClone-{n:fixed=3}" `
-Persistence NonPersistent -deletePolicy DeleteOnUse `
-VmFolderPath "/LinuxVDI/vm/FullClone" `
-ResourcePoolPath "/LinuxVDI/host/LinuxVDICluster/Resources" `
-templatePath "/LinuxVDI/vm/LinuxTemplate" `
-dataStorePaths "/LinuxVDI/host/LinuxVDICluster/datastore" `
-customizationSpecName "linux-spec" `
-maximumCount 100
```

デスクトップ プールの資格を追加または削除

- LinuxDesktop に、ドメイン mydomain.org のドメイン ユーザー グループに付与します。

```
Add-PoolEntitlement -Pool_id LinuxDesktop -Sid (Get-User -Name "domain user" -Domain "mydomain.org").sid
```

- LinuxDesktop から、mydomain.org ドメインのドメイン ユーザー グループの資格を取り外してください。

```
Remove-PoolEntitlement -Pool_id LinuxDesktop -Sid (Get-User -Name "domain user" -Domain "mydomain.org").sid
```

専用デスクトップ プールの仮想マシンに、または仮想マシンからユーザーを割り当てるか削除します

- 専用デスクトップ プールにある、LinuxVM-01 仮想マシンに myuser ユーザーを割り当てます。

```
Update-UserOwnership -Machine_id (Get-DesktopVM -Name "LinuxVM-01").machine_id -Sid (Get-User -Name "myuser" | Where-Object {$_.cn -eq "myuser"}).sid
```

- 専用デスクトップ プールにある、LinuxVM-01 仮想マシンから myuser ユーザーを削除します。

```
Remove-UserOwnership -Machine_id (Get-DesktopVM -Name "LinuxVM-01").machine_id
```

デスクトップ接続をログオフ

- myuser のデスクトップ セッションからログオフします。

```
Get-RemoteSession -Username "mydomain.org\myuser" | Send-SessionLogoff
```

ブローカ PowerCLI cmdlet の詳細については、『View Integration』の「View PowerCLI の 使用」を参照してください。

手動デスクトップ プールのための Horizon 7 の一括デプロイ

8

View Administrator では、Linux ではなく Windows デスクトップ マシンのプールを自動的に作成できます。ただし、Linux デスクトップ マシンのプールのデプロイを自動化するスクリプトを開発できます。

提供されているサンプル スクリプトは、例示のみを目的としています。ユーザーがサンプル スクリプトを使用するときに発生する可能性がある問題について、VMware は一切責任を負いません。

この章には、次のトピックが含まれています。

- [Linux デスクトップの一括デプロイの概要](#)
- [Linux デスクトップの一括アップグレードの概要](#)
- [Linux デスクトップ マシンのクローンを作成するために仮想マシン テンプレートを作成する](#)
- [Linux デスクトップを展開するサンプル PowerCLI スクリプトの入力ファイル](#)
- [Linux 仮想マシンのクローンを作成するサンプル スクリプト](#)
- [クローン作成した仮想マシンを Active Directory ドメインに参加させるサンプル スクリプト](#)
- [SSH を使用してクローン作成した仮想マシンを Active Directory ドメインに参加させるサンプル スクリプト](#)
- [Linux 仮想マシンに構成ファイルをアップロードするサンプル スクリプト](#)
- [SSH を使用して Linux 仮想マシンに構成ファイルをアップロードするサンプル スクリプト](#)
- [Linux デスクトップ マシンで Horizon Agent をアップグレードするサンプル スクリプト](#)
- [SSH を使用して Linux 仮想マシンで Horizon Agent をアップグレードするサンプル スクリプト](#)
- [Linux 仮想マシンで操作を実行するサンプル スクリプト](#)

Linux デスクトップの一括デプロイの概要

Linux 版手動デスクトップ一括デプロイには、いくつかの手順があります。多数のデスクトップをデプロイする予定がある場合、PowerCLI スクリプトを使用していくつかの手順を自動化できます。

一部の操作では、Linux マシン上で PowerCLI または SSH によってコマンドを実行することを選択できます。次の表では、2 つの手法の違いについて説明します。

PowerCLI	SSH
追加のツールのインストールは不要。	<ul style="list-style-type: none"> ■ Ubuntu の場合は、コマンド <code>sudo apt-get install openssh-server</code> を使用して SSH サーバをインストールする必要がある。RHEL と CentOS の場合は、<code>openssh-server</code> がデフォルトでインストールされているが、ファイアウォール設定で <code>ssh</code> が許可されていることを確認する必要がある。 ■ SSH クライアントアプリケーションの <code>pscp.exe</code> と <code>plink.exe</code> をダウンロードし、PowerCLI スクリプトと同じフォルダに配置する必要がある。
ファイルのアップロードとコマンド実行は遅い。	ファイルのアップロードとコマンド実行は速い。
ESXi ホストの管理者認証情報を入力する必要がある。	ESXi ホストの管理者認証情報を入力する必要はない。
管理者パスワード（スクリプトを実行して Horizon Agent をインストールする場合）や Active Directory ユーザーのパスワード（スクリプトを実行してドメインに参加する場合）内の特殊文字を処理できない。	管理者パスワード（スクリプトを実行して Horizon Agent をインストールする場合）や Active Directory ユーザーのパスワード（スクリプトを実行してドメインに参加する場合）内の特殊文字を処理できる。

注: PowerCLI ベースのスクリプトおよび SSH ベースのスクリプトは、vCenter Server 管理者と Linux 管理者のパスワード内の特殊文字を処理できます。PowerCLI ベースのスクリプトは ESXi ホスト管理者のパスワード内の特殊文字も処理できます。これらのいずれの場合もエスケープ文字は不要です。

vSphere PowerCLI の詳細については、<https://www.vmware.com/support/developer/PowerCLI> を参照してください。

Linux デスクトップ プールの一括デプロイプロセスでは、次の手順を実行します。

- 1 仮想マシン テンプレートを作成し、仮想マシンで Horizon Agent をインストールします。

[Linux デスクトップ マシンのクローンを作成するために仮想マシン テンプレートを作成する](#)を参照してください。

- 2 ゲスト カスタマイズの仕様を作成します。

vSphere 仮想マシン管理ドキュメントの「vSphere Web Client での Linux 向けカスタマイズ仕様の作成」を参照してください。仕様を作成する場合、次の設定を必ず正しく指定してください。

設定	値
ターゲット仮想マシンの OS	Linux
コンピュータ名	仮想マシン名を使用します。
ドメイン	View 環境のドメインを指定します。
ネットワーク設定	標準ネットワーク設定を使用します。
プライマリ DNS	有効なアドレスを指定します。

注: ゲスト OS のカスタマイズのサポート一覧の詳細については、<http://partnerweb.vmware.com/programs/guestOS/guest-os-customization-matrix.pdf> を参照してください。

- 3 仮想マシンのクローンを作成します。

[Linux 仮想マシンのクローンを作成するサンプル スクリプト](#)を参照してください。

- 4 winbind ソリューションを使用している場合は、クローン作成された仮想マシンを Active Directory (AD) ドメインに参加させます。下の例スクリプトでドメイン参加コマンドを実行したり、テンプレート仮想マシンで構成された `/etc/vmware/viewagent-custom.conf` 内でオプション `RunOnceScript` を使用できます。

クローン作成した仮想マシンを Active Directory ドメインに参加させるサンプル スクリプトまたは SSH を使用してクローン作成した仮想マシンを Active Directory ドメインに参加させるサンプル スクリプトを参照してください。

- 5 仮想マシンの構成オプションを更新します。

Linux 仮想マシンに構成ファイルをアップロードするサンプル スクリプトまたは SSH を使用して Linux 仮想マシンに構成ファイルをアップロードするサンプル スクリプトを参照してください。

- 6 デスクトップ プールを作成します。

Linux 版手動デスクトップ プールの作成を参照してください。

仮想マシンのパワーオン、シャットダウン、再起動、または削除などの操作を実行するサンプル スクリプトについては、Linux 仮想マシンで操作を実行するサンプル スクリプトを参照してください。このスクリプトを使用して、vCenter Server から仮想マシンを削除できます。

Linux デスクトップの一括アップグレードの概要

Linux 版手動デスクトップの一括アップグレードには、いくつかの手順があります。PowerCLI スクリプトの使用により手順のいくつかを自動化できます。

非管理対象デスクトップの一括アップグレード

非管理対象仮想マシンを、管理対象または非管理対象仮想マシンへ一括アップグレードするには、サンプルのアップグレード スクリプトを使用して新しい Horizon Agent を既存の仮想マシンにアップロードし、アップグレード コマンドを実行する必要があります。

- 非管理対象仮想マシンを保存すると、既存のデスクトップ プールを再使用できます。
- 非管理対象仮想マシンから管理対象仮想マシンへアップグレードする場合は、既存のデスクトップ プールを削除し、新しいデスクトップ プールを作成する必要があります。詳細については、Linux 仮想マシンでの Horizon Agent のアップグレードを参照してください。

管理対象デスクトップの一括アップグレード

管理対象仮想マシンを一括アップグレードするには、次の方法のいずれかを選択します。

方法	説明
テンプレート仮想マシンで、新しい Horizon Agent をインストールするかアップグレードして、スナップショットを作成してください。	<ul style="list-style-type: none"> ■ ユーザー データとプロファイルが NFS サーバのような共有サーバに置かれられない限り、既存の仮想マシンが削除されるため、ユーザー データとプロファイルが失われます。 ■ 仮想マシンの交換後は、View Administrator の仮想マシンの状態が見つからない場合があります。これを修復するにはブローカー サービスを再起動する必要があります。 ■ リンク クローンを使用すると、この方法は各仮想マシンでのデータの重複を回避できます。
アップグレードのサンプル スクリプトを使用して、新しい Horizon Agent を既存の仮想マシンにアップロードを行い、アップグレード コマンドを実行してください。	<ul style="list-style-type: none"> ■ ユーザー データとプロファイルが保存されます。 ■ リンク クローンを使用すると、この方法は各仮想マシンでのデータの重複を招きます。

Linux デスクトップ マシンのクローンを作成するために仮想マシン テンプレートを作成する

仮想マシンのクローンを作成する前に、クローンの基準となる仮想マシン テンプレートを作成する必要があります。

前提条件

- デプロイする環境がサポートする Linux デスクトップの要件を満たしていることを確認します。 [Horizon 7 for Linux のシステム要件](#)を参照してください。
- vCenter Server で仮想マシンを作成し、ゲスト OS をインストールする手順について理解しておきます。View でのデスクトップ プールとアプリケーション プールの設定 ドキュメントの「仮想マシンの作成および準備」を参照してください。
- 仮想マシンで使用するモニタについて推奨されるビデオ メモリ (vRAM) の値を理解しておきます。 [Horizon 7 for Linux のシステム要件](#)を参照してください。
- Active Directory 統合の手順について理解しておきます。 [3 章 Linux デスクトップの Active Directory 統合のセットアップ](#)を参照してください。
- Linux で Horizon Agent をインストールする手順をよく理解しておいてください。「Horizon Agent のインストールと Linux デスクトップの管理」の章を参照してください。
- 必要な場合は、View 構成ファイルを使用してオプションを構成する手順について理解しておきます。 [6 章 Linux デスクトップの構成オプション](#)を参照してください。
- グラフィックスをセットアップする予定がある場合は、その手順について理解しておきます。 [4 章 Linux デスクトップのグラフィックスのセットアップ](#)を参照してください。

手順

- 1 vSphere Web Client または vSphere Client で新しい仮想マシンを作成します。

2 カスタム構成オプションを構成します。

- a 仮想マシンを右クリックし、[設定の編集] をクリックします。
- b vCPU の数と vMemory のサイズを指定します。

推奨値については、お使いの Linux ディストリビューションのインストール ガイドのガイドラインに従ってください。

たとえば、Ubuntu 12.04 では、2048 MB の vMemory と 2 台の vCPU を構成することが推奨されています。

- c [ビデオ カード] を選択して、ディスプレイの数とビデオ メモリ (vRAM) の合計を指定します。

VMware のドライバを使用し、2D や vSGA を使用する仮想マシンについては、vSphere Web Client で vRAM のサイズを設定します。vRAM のサイズは、NVIDIA のドライバを使用する vDGA や NVIDIA GRID vGPU マシンには影響しません。

推奨値については、Horizon 7 for Linux のシステム要件のガイドラインに従ってください。ビデオ メモリ 計算ツールは使用しないでください。

- 3 仮想マシンをパワーオンして、Linux ディストリビューションをインストールします。
- 4 たとえば、ViewUser など root 権限のあるユーザーを作成します。このユーザーは、Horizon Agent のインストールとアンインストールにのみ使用されます。
- 5 `/etc/sudoers` を編集して、行 `ViewUser ALL=(ALL) NOPASSWD:ALL` を追加します。

`/etc/sudoers` のこの行では、ViewUser として `sudo` を実行するためにパスワードは必要ありません。この章で説明しているようにサンプル スクリプトを実行して Horizon Agent をインストールする場合、入力として ViewUser を指定します。

- 6 Linux ディストリビューションが RHEL、CentOS、または NeoKylin である場合、`/etc/sudoers` を編集して、次の行をコメントアウトします。

```
Defaults requiretty
Defaults !visiblepw
```

- 7 Linux ディストリビューションが RHEL 7、CentOS 7、または SLES 12 ではない場合、VMware Tools をインストールします。

RHEL 7、CentOS 7、および SLES 12 には、デフォルトで >Open VM Tools がインストールされています。

- 8 Linux ディストリビューションが RHEL 7、CentOS 7、または SLES 12 の場合、`deployPkg` プラグインをインストールします。

操作手順については、<http://kb.vmware.com/kb/2075048> を参照してください。

- 9 RHEL と CentOS の場合は、[ネットワーク接続] 設定の [自動接続] を有効にします。
- 10 Active Directory 統合タスクを実行します。
- 11 グラフィックスをセットアップする手順を実行します。

12 Horizon Agent をインストールします。

```
sudo ./install_viewagent.sh -A yes
```

13 View 構成ファイルを使用して追加構成を実行します。

14 仮想マシンをシャットダウンして、スナップショットを作成します。

Linux デスクトップを展開するサンプル PowerCLI スクリプトの入力ファイル

Linux デスクトップを展開するサンプル PowerCLI スクリプトは、デスクトップ マシンに関する情報を含む 1 つの入力ファイルを読み取ります。

入力ファイルのタイプは csv であり、次の情報を含みます。

- デスクトップ仮想マシンの名前
- 親仮想マシンの名前
- ゲスト カスタマイズの仕様
- クローン作成されたデスクトップ マシンが存在するデータストア
- デスクトップ マシンをホストする ESXi サーバ
- クローン作成に使用される親仮想マシンのスナップショット
- 存在している場合、デスクトップ仮想マシンを削除するかどうかを示すフラグ

次の例は、入力ファイルに含まれる可能性がある情報を示しています。

```
VMName,Parentvm,CustomSpec,Datastore,Host,FromSnapshot,DeleteIfPresent
linux-001,Ubuntu1204x64,linuxagent,datastore1,10.117.44.172,snapshot1,TRUE
linux-002,Ubuntu1204x64,linuxagent,datastore1,10.117.44.172,snapshot1,TRUE
linux-003,Ubuntu1204x64,linuxagent,datastore1,10.117.44.172,snapshot1,TRUE
linux-004,Ubuntu1204x64,linuxagent,datastore1,10.117.44.172,snapshot1,TRUE
linux-005,Ubuntu1204x64,linuxagent,datastore1,10.117.44.172,snapshot1,TRUE
```

サンプル スクリプトでは、入力ファイルの名前が CloneVMs.csv であり、このファイルがスクリプトと同じフォルダにある配置されていることを前提としています。

Linux 仮想マシンのクローンを作成するサンプル スクリプト

次のサンプル スクリプトをカスタマイズして使用し、任意の数の仮想マシン (VM) のクローンを作成できます。

改ページせずにスクリプトの内容をコピーして貼り付けるには、このトピックの HTML 版を使用します。HTML 版は https://www.vmware.com/support/pubs/view_pubs.html にある Horizon 7 のドキュメントのページから入手できます。

スクリプト入力

このスクリプトは、[Linux デスクトップを展開するサンプル PowerCLI スクリプトの入力ファイル](#)で説明しているように 1 つの入力ファイルを読み取ります。また、このスクリプトは、次の情報をインタラクティブに確認します。

- vCenter Server の IP アドレス
- vCenter Server 管理者のログイン名
- vCenter Server 管理者のパスワード
- クローンのタイプ。リンク クローンまたは完全クローンになります
- vSphere 仮想マシン コンソールを無効にするかどうか

スクリプトのコンテンツ

```
<#
Create Clones from a Master VM

The Tool supports creation of Full clone and linked clone from Master VM.
The parent VM is required for the linked-clone to work and the parent VMs file cannot be renamed or
moved.
#>
#----- Functions -----
function GetInput
{
    Param($prompt, $IsPassword = $false)
    $prompt = $prompt + ": "
    Write-Host $prompt -NoNewLine
    [Console]::ForegroundColor = "Blue"
    if ($IsPassword)
    {
        $input = Read-Host -AsSecureString
        $input =
[Runtime.InteropServices.Marshal]::PtrToStringAuto([Runtime.InteropServices.Marshal]::SecureStringToBS
TR($input))
    }
    else
    {
        $input = Read-Host
    }

    [Console]::ResetColor()
    return $input
}

function IsVMExists ()
{
    Param($VMExists)
    Write-Host "Checking if the VM $VMExists already Exists"
    [bool]$Exists = $false

    #Get all VMS and check if the VMs is already present in VC
    $listvm = Get-vm
```

```

    foreach ($lvm in $listvm)
    {
        if($VMExists -eq $lvm.Name )
        {
            $Exists = $true
        }
    }
    return $Exists
}
function Disable_VM_Console()
{
    Param($VMToDisableConsole)
    $vmConfigSpec = New-Object VMware.Vim.VirtualMachineConfigSpec
    $extra = New-Object VMware.Vim.optionvalue
    $extra.Key="RemoteDisplay.maxConnections"
    $extra.Value="0"
    $vmConfigSpec.extraconfig += $extra
    $vm = Get-VM $VMToDisableConsole | Get-View
    $vm.ReconfigVM($vmConfigSpec)
}

function Delete_VM()
{
    Param($VMToDelete)
    Write-Host "Deleting VM $VMToDelete"
    Get-VM $VMToDelete | where { $_.PowerState -eq "PoweredOn" } | Stop-VM -confirm:$false
    Get-VM $VMToDelete | Remove-VM -DeleteFromDisk -confirm:$false
}

#----- Main Script -----

SvcAddress = GetInput -prompt "Your vCenter address" -IsPassword $false
SvcAdmin = GetInput -prompt "Your vCenter admin user name" -IsPassword $false
SvcPassword = GetInput -prompt "Your vCenter admin user password" -IsPassword $true
$cloneType = GetInput -prompt 'Clone Type ("linked" or "full")' -IsPassword $false
$disableVMConsole = GetInput -prompt 'Disable vSphere VM Console ("yes" or "no", recommend "yes")' -
IsPassword $false
"-----"
$csvFile = '.\CloneVMs.csv'

# Check that user passed only linked or full clone
if (($CloneType.length > 0) -and ($CloneType -ne "linked" -or $CloneType -ne "full"))
{
    write-host -ForegroundColor Red "Clone type supports only 'linked' or 'full' (case sensitive)"
    exit
}
if (($disableVMConsole.length > 0) -and ($disableVMConsole -ne "yes" -or $disableVMConsole -ne "no"))
{
    write-host -ForegroundColor Red "Disable vSphere VM Console supports only 'yes' or 'no' (case
sensitive)"
    exit
}

#check if file exists

```

```

if (!(Test-Path $csvFile))
{
    write-host -ForegroundColor Red "CSV File $CSVFile not found"
    exit
}

# Connect to the VC (Parameterize VC)
#Connect to vCenter
$VC_Conn_State = Connect-VIServer $vcAddress -user $vcAdmin -password $vcPassword
if([string]::IsNullOrEmpty($VC_Conn_State))
{
    Write-Host 'Exit since failed to login vCenter'
    exit
}
else
{
    Write-Host 'vCenter is connected'
}

#Read input CSV file
$csvData = Import-CSV $csvFile
#$csvData = Import-CSV $csvFile -
header("VMName","Parentvm","CustomSpec","Datastore","Host","FromSnapshot","DeleteIfPresent")
foreach ($line in $csvData)
{
    "`n-----"
    $VMName = $line.VMName
    write-host -ForegroundColor Yellow "VM: $VMName`n"

    $destVMName=$line.VMName
    $srcVM = $line.Parentvm
    $cSpec = $line.CustomSpec
    $targetDSName = $line.Datastore
    $destHost = $line.Host
    $srcSnapshot = $line.FromSnapshot
    $deleteExisting = $line.DeleteIfPresent
    if (IsVMExists ($destVMName))
    {
        Write-Host "VM $destVMName Already Exists in VC $vcAddress"
        if($deleteExisting -eq "TRUE")
        {
            Delete_VM ($destVMName)
        }
        else
        {
            Write-Host "Skip clone for $destVMName"
            continue
        }
    }
    $vm = get-vm $srcvm -ErrorAction Stop | get-view -ErrorAction Stop
    $cloneSpec = new-object VMware.VIM.VirtualMachineCloneSpec
    $cloneSpec.Location = new-object VMware.VIM.VirtualMachineRelocateSpec
    if ($CloneType -eq "linked")
    {
        $cloneSpec.Location.DiskMoveType =

```

```
[VMware.VIM.VirtualMachineRelocateDiskMoveOptions]::createNewChildDiskBacking
}
Write-Host "Using Datastore $targetDSName"
$newDS = Get-Datastore $targetDSName | Get-View
$cloneSpec.Location.Datastore = $newDS.summary.Datastore
Set-VM -vm $srcVM -snapshot (Get-Snapshot -vm $srcVM -Name $srcSnapshot) -confirm:$false
$cloneSpec.Snapshot = $vm.Snapshot.CurrentSnapshot
$cloneSpec.Location.Host = (get-vmhost -Name $destHost).Extensiondata.MoRef
$cloneSpec.Location.Pool = (Get-ResourcePool -Name Resources -Location (Get-VMHost -Name
$destHost)).Extensiondata.MoRef
# Start the Clone task using the above parameters
$task = $vm.CloneVM_Task($vm.parent, $destVMName, $cloneSpec)
# Get the task object
$task = Get-Task | where { $_.id -eq $task }
#Wait for the taks to Complete
Wait-Task -Task $task

$newvm = Get-vm $destVMName
$customSpec = Get-OSCustomizationSpec $cSpec
Set-vm -OSCustomizationSpec $cSpec -vm $newvm -confirm:$false
if ($disableVMConsole -eq "yes")
{
    Disable_VM_Console($destVMName)
}
# Start the VM
Start-VM $newvm
}
Disconnect-VIServer $vcAddress -Confirm:$false
exit
```

スクリプトの実行

このスクリプトを実行すると、次のメッセージが表示されます。

```
PowerCLI C:\scripts> .\CloneVMs.ps1
Your vCenter address: 10.117.44.17
Your vCenter admin user name: administrator
Your vCenter admin user password: *****
Clone Type<"linked" or "Full"> : linked
Disable vSphere VM Console ("yes" or "no", recommend "yes") : yes
```

クローン作成プロセスにかかる時間は、デスクトップマシンの数によって異なり、数分から数時間になります。プロセスが完了したことを確認するには、vSphere Client で、最後のデスクトップ仮想マシンがパワーオンされていること、一意のホスト名が付いていること、VMware Tools が動作していることを確認します。

クローン作成した仮想マシンを Active Directory ドメインに参加させるサンプル スクリプト

次のサンプル スクリプトをカスタマイズして使用し、クローン作成した仮想マシン (VM) を Active Directory (AD) ドメインに参加させることができます。

Active Directory 統合に Winbind ソリューションを使用する場合は、このスクリプトを実行する必要があります。クローン作成した仮想マシンでは、ドメインに参加する手順がエラーになるためです。このスクリプトでは、ドメインに参加するためのコマンドが各仮想マシンで実行されます。OpenLDAP ソリューションを使用する場合、このスクリプトを実行する必要はありません。

改ページせずにスクリプトの内容をコピーして貼り付けるには、このトピックの HTML 版を使用します。HTML 版は https://www.vmware.com/support/pubs/view_pubs.html にある Horizon 7 のドキュメントのページから入手できます。

スクリプト入力

このスクリプトは、[Linux デスクトップを展開するサンプル PowerCLI スクリプトの入力ファイル](#)で説明しているように 1 つの入力ファイルを読み取ります。また、このスクリプトは、次の情報をインタラクティブに確認します。

- vCenter Server の IP アドレス
- vCenter Server 管理者のログイン名
- vCenter Server 管理者のパスワード
- ESXi ホスト管理者のログイン名
- ESXi ホスト管理者のパスワード
- Linux 仮想マシンのユーザー ログイン名
- Linux 仮想マシンのユーザー パスワード
- マシンをドメインに参加させる許可を受けた Active Directory ユーザーのログイン名
- 許可された Active Directory ユーザーのパスワード

スクリプトのコンテンツ

```
<#
.SYNOPSIS
run command "sudo /usr/bin/net ads join"

.DESCRIPTION
The tool is to run the command "sudo /usr/bin/net ads join" to join Linux to AD

.NOTES
#>
#----- Functions -----
function GetInput
{
    Param($prompt, $IsPassword = $false)
    $prompt = $prompt + ": "
    Write-Host $prompt -NoNewLine
    [Console]::ForegroundColor = "Blue"
    if ($IsPassword)
    {
        $input = Read-Host -AsSecureString
        $input =
[Runtime.InteropServices.Marshal]::PtrToStringAuto([Runtime.InteropServices.Marshal]::SecureStringToBS
```

```

TR($input))
}
else
{
    $input = Read-Host
}

[Console]::ResetColor()
return $input
}
#----- Handle input -----
"-----"
$vcAddress = GetInput -prompt "Your vCenter address" -IsPassword $false
$vcAdmin = GetInput -prompt "Your vCenter admin user name" -IsPassword $false
$vcPassword = GetInput -prompt "Your vCenter admin user password" -IsPassword $true
"-----"
$hostAdmin = GetInput -prompt 'Your ESXi host admin user name, such as root' -IsPassword $false
$hostPassword = GetInput -prompt "Your ESXi admin user password" -IsPassword $true
"-----"
$guestUser = GetInput -prompt 'Your VM guest OS user name' -IsPassword $false
$guestPassword = GetInput -prompt 'Your VM guest OS user password' -IsPassword $true
"-----"
$adUser = GetInput -prompt 'Type the AD user name to join the AD' -IsPassword $false
""
"Please type the AD user password."
"Please note that special character in password may not work with the script"
$adUserPassword = GetInput -prompt 'Your AD user password' -IsPassword $true
"-----"

#$csvFile = Read-Host 'Csv File '
$csvFile = '.\CloneVMs.csv'

#----- Main Script -----

#Connect to vCenter
#Connect to vCenter
$VC_Conn_State = Connect-VIServer $vcAddress -user $vcAdmin -password $vcPassword
if([string]::IsNullOrEmpty($VC_Conn_State))
{
    Write-Host 'Exit since failed to login vCenter'
    exit
}
else
{
    Write-Host 'vCenter is connected'
}

#Read input CSV file
$csvData = Import-CSV $csvFile

$destFolder = "/home/$guestUser/"

#Handle VMs one by one
foreach ($line in $csvData)
{

```

```

" `n-----"
$VMName = $line.VMName
write-host -ForegroundColor Yellow "VM: $VMName`n"

$cmd = "sudo /usr/bin/net ads join -U $adUser%$adUserPassword"
Write-Host "Run cmd 'sudo /usr/bin/net ads join' in VM '$VMName' with user '$guestUser'"
Invoke-VMScript -HostUser $hostAdmin -HostPassword $hostPassword -VM $VMName -GuestUser
$guestUser -GuestPassword $guestPassword -Confirm:$false -ScriptType Bash -ScriptText $cmd
}

Disconnect-VIServer $vcAddress -Confirm:$false
exit

```

スクリプトの実行

このスクリプトを実行すると、次のメッセージが表示されます。

```

PowerCLI C:\scripts> .\ClonedVMs_JoinDomain.ps1

-----
Your vCenter address: 10.117.44.17
Your vCenter admin user name: administrator
Your vCenter admin user password: *****

-----
Your ESXi host admin user name, such as root: root
Your ESXi host admin user password: *****

-----
Your VM guest OS user name: ViewUser
Your VM guest OS user password: *****

-----
Type the AD user name to join the AD: viewadmin
Please type the AD user password.
Please note that special character in password may not work with the script.
Your AD user password: *****

```

SSH を使用してクローン作成した仮想マシンを Active Directory ドメインに参加させるサンプル スクリプト

次のサンプル スクリプトをカスタマイズして使用し、クローン作成した仮想マシン (VM) を Active Directory (AD) ドメインに参加させることができます。このスクリプトでは SSH を使用して、Linux 仮想マシンでコマンドを実行します。

Active Directory 統合に Winbind ソリューションを使用する場合は、このスクリプトを実行する必要があります。クローン作成した仮想マシンでは、ドメインに参加する手順がエラーになるためです。このスクリプトでは、ドメインに参加するためのコマンドが各仮想マシンで実行されます。OpenLDAP ソリューションを使用する場合、このスクリプトを実行する必要はありません。

改ページせずにスクリプトの内容をコピーして貼り付けるには、このトピックの HTML 版を使用します。HTML 版は https://www.vmware.com/support/pubs/view_pubs.html にある Horizon 7 のドキュメントのページから入手できます。

スクリプト入力

このスクリプトは、[Linux デスクトップを展開するサンプル PowerCLI スクリプトの入力ファイル](#)で説明しているように 1 つの入力ファイルを読み取ります。また、このスクリプトは、次の情報をインタラクティブに確認します。

- vCenter Server の IP アドレス
- vCenter Server 管理者のログイン名
- vCenter Server 管理者のパスワード
- Linux 仮想マシンのユーザー ログイン名
- Linux 仮想マシンのユーザー パスワード
- マシンをドメインに参加させる許可を受けた Active Directory ユーザーのログイン名
- 許可された Active Directory ユーザーのパスワード

スクリプトのコンテンツ

```
<#
.SYNOPSIS
run command "sudo /usr/bin/net ads join" via SSH

.DESCRIPTION
The tool is to run the command "sudo /usr/bin/net ads join" to join Linux machine to AD via SSH

.NOTES
#>
#----- Functions -----
function GetInput
{
    Param($prompt, $IsPassword = $false)
    $prompt = $prompt + ": "
    Write-Host $prompt -NoNewLine
    [Console]::ForegroundColor = "Blue"
    if ($IsPassword)
    {
        $input = Read-Host -AsSecureString
        $input =
[Runtime.InteropServices.Marshal]::PtrToStringAuto([Runtime.InteropServices.Marshal]::SecureStringToBS
TR($input))
    }
    else
    {
        $input = Read-Host
    }

    [Console]::ResetColor()
    return $input
}

function Check_SSH_Client
{
    Param($IsPlink, $IsPSCP)
```

```

if ($IsPlink)
{
    if (Test-Path ".\plink.exe")
    {
        write-host -ForegroundColor Yellow 'SSH client "plink.exe" found'
    }
    else
    {
        write-host -ForegroundColor Red 'SSH client "plink.exe" not found, please download from
its official web site'
        exit
    }
}
if ($IsPSCP)
{
    if (Test-Path ".\pscp.exe")
    {
        write-host -ForegroundColor Yellow 'SSH client "pscp.exe" found'
    }
    else
    {
        write-host -ForegroundColor Red 'SSH client "pscp.exe" not found, please download from its
official web site'
        exit
    }
}

function RunCmdViaSSH
{
    Param($VM_Name, $User, $Password, $Cmd, $returnOutput = $false)

    $VM= Get-VM $VM_Name
    $IP = $VM.guest.IPAddress[0]
    write-host "Run cmd on $VM_Name ($IP)"
    if($returnOutput)
    {
        $command = "echo yes | .\plink.exe -ssh -l $user -pw $password $IP " + "'" + $cmd + "'"
        $output = Invoke-Expression $command
        return $output
    }
    else
    {
        {
            echo yes | .\plink.exe -ssh -l $user -pw $password $IP "$cmd"
        }
    }
}

function UploadFileViaSSH
{
    Param($VM_Name, $User, $Password, $LocalPath, $DestPath)

    $VM= Get-VM $VM_Name
    $IP = $VM.guest.IPAddress[0]
    $command = "echo yes | .\pscp.exe -l $User -pw $Password $LocalPath $IP" + ":" + "$DestPath"

```

```

    write-host "Upload file: $command"
    Invoke-Expression $command
}

#----- Handle input -----
"-----"
Check_SSH_Client -IsPlink $true -IsPSCP $false
"-----"
$vcAddress = GetInput -prompt "Your vCenter address" -IsPassword $false
$vcAdmin = GetInput -prompt "Your vCenter admin user name" -IsPassword $false
$vcPassword = GetInput -prompt "Your vCenter admin user password" -IsPassword $true
"-----"
$guestUser = GetInput -prompt 'Your VM guest OS user name' -IsPassword $false
$guestPassword = GetInput -prompt 'Your VM guest OS user password' -IsPassword $true
"-----"
$adUser = GetInput -prompt 'Type the AD user name to join the AD' -IsPassword $false
""
`nPlease type the AD user password."
[Console]::ForegroundColor = "Yellow"
"Plase note that special character should be escaped. For example, $ should be \$\"
[Console]::ResetColor()
$adUserPassword = GetInput -prompt 'Your AD user password' -IsPassword $true
"-----"

#$csvFile = Read-Host 'Csv File '
$csvFile = '.\CloneVMs.csv'

#----- Main Script -----

#Connect to vCenter
$VC_Conn_State = Connect-VIServer $vcAddress -user $vcAdmin -password $vcPassword
if([string]::IsNullOrEmpty($VC_Conn_State))
{
    Write-Host 'Exit since failed to login vCenter'
    exit
}
else
{
    Write-Host 'vCenter is connected'
}

#Read input CSV file
$csvData = Import-CSV $csvFile

$destFolder = "/home/$guestUser/"

#Handle VMs one by one
foreach ($line in $csvData)
{
    "-----"
    $VMName = $line.VMName
    write-host -ForegroundColor Yellow "VM: $VMName`n"

    $cmd = "sudo /usr/bin/net ads join -U $adUser%$adUserPassword"
    Write-Host "Run cmd 'sudo /usr/bin/net ads join' in VM '$VMName' with user '$guestUser'"
}

```

```

    RunCmdViaSSH -VM_Name $VMName -User $guestUser -Password $guestPassword -Cmd $cmd
}

Disconnect-VIServer $vcAddress -Confirm:$false
exit

```

スクリプトの実行

このスクリプトを実行すると、次のメッセージが表示されます。

```

PowerCLI C:\scripts> .\ClonedVMs_JoinDomain_SSH.ps1

-----
Your vCenter address: 10.117.44.17
Your vCenter admin user name: administrator
Your vCenter admin user password: *****
-----
Your VM guest OS user name: ViewUser
Your VM guest OS user password: *****
-----
Type the AD user name to join the AD: viewadmin
Please type the AD user password.
Please note that special character should be escaped. For example, $ should be \$
Your AD user password: *****

```

Linux 仮想マシンに構成ファイルをアップロードするサンプル スクリプト

次のサンプル スクリプトをカスタマイズして使用し、構成ファイル config と viewagent-custom.conf を複数の Linux 仮想マシン (VM) にアップロードできます。

改ページせずにスクリプトの内容をコピーして貼り付けるには、このトピックの HTML 版を使用します。HTML 版は https://www.vmware.com/support/pubs/view_pubs.html にある Horizon 7 のドキュメントのページから入手できます。

スクリプト入力

このスクリプトは、[Linux デスクトップを展開するサンプル PowerCLI スクリプトの入力ファイル](#)で説明しているように 1 つの入力ファイルを読み取ります。また、このスクリプトは、次の情報をインタラクティブに確認します。

- vCenter Server の IP アドレス
- vCenter Server 管理者のログイン名
- vCenter Server 管理者のパスワード
- ESXi ホスト管理者のログイン名
- ESXi ホスト管理者のパスワード
- Linux 仮想マシンのユーザー ログイン名
- Linux 仮想マシンのユーザー パスワード

スクリプトのコンテンツ

```
<#
Upload the configuration files config and viewagent-custom.conf to Linux VMs
#>
#----- Functions -----
function GetInput
{
    Param($prompt, $IsPassword = $false)
    $prompt = $prompt + ": "
    Write-Host $prompt -NoNewLine
    [Console]::ForegroundColor = "Blue"
    if ($IsPassword)
    {
        $input = Read-Host -AsSecureString
        $input =
[Runtime.InteropServices.Marshal]::PtrToStringAuto([Runtime.InteropServices.Marshal]::SecureStringToBS
TR($input))
    }
    else
    {
        $input = Read-Host
    }

    [Console]::ResetColor()
    return $input
}

#----- Handle Input -----
"-----"
write-host -ForegroundColor Blue 'Please ensure your config file and viewagent-custom.conf file are
in current working directory'
$vcAddress = GetInput -prompt "Your vCenter address" -IsPassword $false
$vcAdmin = GetInput -prompt "Your vCenter admin user name" -IsPassword $false
$vcPassword = GetInput -prompt "Your vCenter admin user password" -IsPassword $true
"-----"
$hostAdmin = GetInput -prompt 'Your ESXi host admin user name, such as root' -IsPassword $false
$hostPassword = GetInput -prompt "Your ESXi admin user password" -IsPassword $true
"-----"
$guestUser = GetInput -prompt 'Your VM guest OS user name' -IsPassword $false
$guestPassword = GetInput -prompt 'Your VM guest OS user password' -IsPassword $true
"-----"

$csvFile = '.\CloneVMs.csv'
$setConfig = $false
$setCustomConf = $false
$config_File = "config"
$customConf_File = "viewagent-custom.conf"

#check if config file exists
if(Test-Path $config_File)
{
    $setConfig = $true
    write-host -ForegroundColor Yellow '"config" file found'
```

```

}
else
{
    write-host -ForegroundColor Yellow '"config" file not found, skip it'
}

if(Test-Path $customConf_File)
{
    $setCustomConf = $true
    write-host -ForegroundColor Yellow '"viewagent-custom.conf" file found'
}
else
{
    write-host -ForegroundColor Yellow '"viewagent-custom.conf" file not found, skip it'
}

if (($setConfig -eq $false)-AND ($setCustomConf -eq $false))
{
    write-host -ForegroundColor Red 'Both file not found, exit'
    exit
}

#Connect to vCenter
$VC_Conn_State = Connect-VIServer $vcAddress -user $vcAdmin -password $vcPassword
if([string]::IsNullOrEmpty($VC_Conn_State))
{
    Write-Host 'Exit since failed to login vCenter'
    exit
}
else
{
    Write-Host 'vCenter is connected'
}

#Read input CSV file
$csvData = Import-CSV $csvFile

$destFolder = "/home/$guestUser/"

#Handle VMs one by one
foreach ($line in $csvData)
{
    "`n-----"
    $VMName = $line.VMName
    write-host -ForegroundColor Yellow "VM: $VMName`n"

    #Try to delete the configuration file from home folder on destination VM
    $cmd = "rm -rf config viewagent-custom.conf"
    Write-Host "Run cmd '$cmd' in VM '$VMName' with user '$guestUser'"
    Invoke-VMScript -HostUser $hostAdmin -HostPassword $hostPassword -VM $VMName -GuestUser
$guestUser -GuestPassword $guestPassword -Confirm:$false -ScriptType Bash -ScriptText $cmd

    if ($setConfig)
    {

```

```

Write-Host "Upload File '$config_File' to '$destFolder' of VM '$VMName' with user '$guestUser'"
Copy-VMGuestFile -HostUser $hostAdmin -HostPassword $hostPassword -VM $VMName -GuestUser
$guestUser -GuestPassword $guestPassword -Confirm:$false -LocalToGuest -Destination $destFolder -
Source $config_File

$cmd = "sudo mv ./ $config_File /etc/vmware/";
Write-Host "Move configuraton file: $cmd"
Invoke-VMScript -HostUser $hostAdmin -HostPassword $hostPassword -VM $VMName -GuestUser
$guestUser -GuestPassword $guestPassword -Confirm:$false -ScriptType Bash -ScriptText $cmd
}

if ($setCustomConf)
{
Write-Host "Upload File '$customConf_File' to '$destFolder' of VM '$VMName' with user
'$guestUser'"
Copy-VMGuestFile -HostUser $hostAdmin -HostPassword $hostPassword -VM $VMName -GuestUser
$guestUser -GuestPassword $guestPassword -Confirm:$false -LocalToGuest -Destination $destFolder -
Source $customConf_File

$cmd = "sudo mv ./ $customConf_File /etc/vmware/";
Write-Host "Move configuraton file: $cmd"
Invoke-VMScript -HostUser $hostAdmin -HostPassword $hostPassword -VM $VMName -GuestUser
$guestUser -GuestPassword $guestPassword -Confirm:$false -ScriptType Bash -ScriptText $cmd
}
}
Disconnect-VIServer $vcAddress -Confirm:$false
exit

```

スクリプトの実行

このスクリプトを実行すると、次のメッセージが表示されます。

```

PowerCLI C:\scripts> .\UpdateOptionFile.ps1
-----
Please ensure your config file and view-agent.conf file are in current working directory.
Your vCenter address: 10.117.44.17
Your vCenter admin user name: administrator
Your vCenter admin user password: *****
-----
Your ESXi host admin user name, such as root: root
Your ESXi host admin user password: *****
-----
Your VM guest OS user name: ViewUser
Your VM guest OS user password: *****

```

SSH を使用して Linux 仮想マシンに構成ファイルをアップロードする サンプル スクリプト

次のサンプル スクリプトをカスタマイズして使用し、構成ファイル config と viewagent-custom.conf を複数の Linux 仮想マシン (VM) にアップロードできます。このスクリプトでは SSH を使用して、Linux 仮想マシンでコマンドを実行します。

改ページせずにスクリプトの内容をコピーして貼り付けるには、このトピックの HTML 版を使用します。HTML 版は https://www.vmware.com/support/pubs/view_pubs.html にある Horizon 7 のドキュメントのページから入手できます。

スクリプト入力

このスクリプトは、[Linux デスクトップを展開するサンプル PowerCLI スクリプトの入力ファイル](#)で説明しているように 1 つの入力ファイルを読み取ります。また、このスクリプトは、次の情報をインタラクティブに確認します。

- vCenter Server の IP アドレス
- vCenter Server 管理者のログイン名
- vCenter Server 管理者のパスワード
- Linux 仮想マシンのユーザー ログイン名
- Linux 仮想マシンのユーザー パスワード

スクリプトのコンテンツ

```
<#
Upload the configuration files config and viewagent-custom.conf to Linux VMs using SSH
#>
#----- Functions -----
function GetInput
{
    Param($prompt, $IsPassword = $false)
    $prompt = $prompt + ": "
    Write-Host $prompt -NoNewLine
    [Console]::ForegroundColor = "Blue"
    if ($IsPassword)
    {
        $input = Read-Host -AsSecureString
        $input =
[Runtime.InteropServices.Marshal]::PtrToStringAuto([Runtime.InteropServices.Marshal]::SecureStringToBS
TR($input))
    }
    else
    {
        $input = Read-Host
    }

    [Console]::ResetColor()
    return $input
}
function Check_SSH_Client
{
    Param($IsPlink, $IsPSCP)
    if ($IsPlink)
    {
        if (Test-Path ".\plink.exe")
        {
            write-host -ForegroundColor Yellow 'SSH client "plink.exe" found'
```



```

    }
    else
    {
        write-host -ForegroundColor Red 'SSH client "plink.exe" not found, please download from
its official web site'
        exit
    }
}
if ($IsPSCP)
{
    if (Test-Path ".\pscp.exe")
    {
        write-host -ForegroundColor Yellow 'SSH client "pscp.exe" found'
    }
    else
    {
        write-host -ForegroundColor Red 'SSH client "pscp.exe" not found, please download from its
official web site'
        exit
    }
}
}

function RunCmdViaSSH
{
    Param($VM_Name, $User, $Password, $Cmd, $returnOutput = $false)

    $VM= Get-VM $VM_Name
    $IP = $VM.guest.IPAddress[0]
    write-host "Run cmd on $VM_Name ($IP)"
    if($returnOutput)
    {
        $command = "echo yes | .\plink.exe -ssh -l $user -pw $password $IP " + '"' + $cmd + '"'
        $output = Invoke-Expression $command
        return $output
    }
    else
    {
        echo yes | .\plink.exe -ssh -l $user -pw $password $IP "$cmd"
    }
}

function UploadFileViaSSH
{
    Param($VM_Name, $User, $Password, $LocalPath, $DestPath)

    $VM= Get-VM $VM_Name
    $IP = $VM.guest.IPAddress[0]
    $command = "echo yes | .\pscp.exe -l $User -pw $Password $LocalPath $IP" + ":" + "$DestPath"
    write-host "Upload file: $command"
    Invoke-Expression $command
}

#----- Handle Input -----

```

```

"-----"
Check_SSH_Client -IsPlink $true -IsPSCP $true
"-----"

write-host -ForegroundColor Blue 'Please ensure your config file and viewagent-custom.conf file are
in current working directory'
$vcAddress = GetInput -prompt "Your vCenter address" -IsPassword $false
$vcAdmin = GetInput -prompt "Your vCenter admin user name" -IsPassword $false
$vcPassword = GetInput -prompt "Your vCenter admin user password" -IsPassword $true
"-----"

$guestUser = GetInput -prompt 'Your VM guest OS user name' -IsPassword $false
$guestPassword = GetInput -prompt 'Your VM guest OS user password' -IsPassword $true
"-----"

$csvFile = '.\CloneVMs.csv'
$setConfig = $false
$setCustomConf = $false
$config_File = "config"
$customConf_File = "viewagent-custom.conf"

#check if config file exists
if(Test-Path $config_File)
{
    $setConfig = $true
    write-host -ForegroundColor Yellow '"config" file found'
}
else
{
    write-host -ForegroundColor Yellow '"config" file not found, skip it'
}

if(Test-Path $customConf_File)
{
    $setCustomConf = $true
    write-host -ForegroundColor Yellow '"viewagent-custom.conf" file found'
}
else
{
    write-host -ForegroundColor Yellow '"viewagent-custom.conf" file not found, skip it'
}

if (($setConfig -eq $false)-AND ($setCustomConf -eq $false))
{
    write-host -ForegroundColor Red 'Both file not found, exit'
    exit
}

#Connect to vCenter
$VC_Conn_State = Connect-VIServer $vcAddress -user $vcAdmin -password $vcPassword
if([string]::IsNullOrEmpty($VC_Conn_State))
{
    Write-Host 'Exit since failed to login vCenter'
    exit
}
else
{

```

```

Write-Host 'vCenter is connected'
}

#Read input CSV file
$csvData = Import-CSV $csvFile

$destFolder = "/home/$guestUser/"

#Handle VMs one by one
foreach ($line in $csvData)
{
    "`n-----"
    $VMName = $line.VMName
    write-host -ForegroundColor Yellow "VM: $VMName`n"

    #Try to delete the configuration file from home folder on destination VM
    $cmd = "rm -rf config viewagent-custom.conf"
    Write-Host "Run cmd '$cmd' in VM '$VMName' with user '$guestUser'"
    RunCmdViaSSH -VM_Name $VMName -User $guestUser -Password $guestPassword -Cmd $cmd

    if ($setConfig)
    {
        Write-Host "Upload File '$config_File' to '$destFolder' of VM '$VMName' with user '$guestUser'"
        UploadFileViaSSH -VM_Name $VMName -User $guestUser -Password $guestPassword -LocalPath
$config_File -DestPath $destFolder

        $cmd = "sudo mv ./ $config_File /etc/vmware/";
        Write-Host "Move configuraton file: $cmd"
        RunCmdViaSSH -VM_Name $VMName -User $guestUser -Password $guestPassword -Cmd $cmd
    }

    if ($setCustomConf)
    {
        Write-Host "Upload File '$customConf_File' to '$destFolder' of VM '$VMName' with user
'$guestUser'"
        UploadFileViaSSH -VM_Name $VMName -User $guestUser -Password $guestPassword -LocalPath
$customConf_File -DestPath $destFolder

        $cmd = "sudo mv ./ $customConf_File /etc/vmware/";
        Write-Host "Move configuraton file: $cmd"
        RunCmdViaSSH -VM_Name $VMName -User $guestUser -Password $guestPassword -Cmd $cmd
    }
}
Disconnect-VIServer $vcAddress -Confirm:$false
exit

```

スクリプトの実行

このスクリプトを実行すると、次のメッセージが表示されます。

```

PowerCLI C:\scripts> .\UpdateOptionFile.ps1
-----
Please ensure your config file and view-agent.conf file are in current working directory.
Your vCenter address: 10.117.44.17

```

```

Your vCenter admin user name: administrator
Your vCenter admin user password: *****
-----
Your VM guest OS user name: ViewUser
Your VM guest OS user password: *****

```

Linux デスクトップ マシンで Horizon Agent をアップグレードする サンプル スクリプト

次のサンプル スクリプトをカスタマイズして使用し、複数の Linux 仮想マシン (VM) で Horizon Agent をアップグレードできます。

このスクリプトでは、Horizon Agent のインストール前に、各仮想マシンにインストーラの tar ボールをアップロードします。アップロード タスクは、多くの仮想マシンが含まれ、ネットワークのスピードが遅い場合は特に時間がかかることがあります。時間を節約するには、SSH を使用するスクリプトを実行するか、インストーラの tar ボールを共有場所に配置して各仮想マシンで使用できるようにして、ファイルのアップロードを不要にすることができます。

改ページせずにスクリプトの内容をコピーして貼り付けるには、このトピックの HTML 版を使用します。HTML 版は https://www.vmware.com/support/pubs/view_pubs.html にある Horizon 7 のドキュメントのページから入手できます。

スクリプト入力

このスクリプトは、[Linux デスクトップを展開するサンプル PowerCLI スクリプトの入力ファイル](#)で説明しているように 1 つの入力ファイルを読み取ります。また、このスクリプトは、次の情報をインタラクティブに確認します。

- Horizon Agent EULA（エンド ユーザー使用許諾契約書）の承諾
- vCenter Server の IP アドレス
- vCenter Server 管理者のログイン名
- vCenter Server 管理者のパスワード
- ESXi ホスト管理者のログイン名
- ESXi ホスト管理者のパスワード
- Linux ゲスト OS のユーザー ログイン名
- Linux ゲスト OS のユーザー パスワード
- Horizon Agent の tar ボールのパス
- 管理対象の仮想マシンへアップグレード
- スマートカード リダイレクト機能をインストール

スクリプトのコンテンツ

```
<#
Upload the Linux Agent installer tar ball and re-install
#>

#-----
Functions-----
function GetInput
{
    Param($prompt, $IsPassword = $false)
    $prompt = $prompt + ": "
    Write-Host $prompt -NoNewLine
    [Console]::ForegroundColor = "Blue"
    if ($IsPassword)
    {
        $input = Read-Host -AsSecureString
        $input =
[Runtime.InteropServices.Marshal]::PtrToStringAuto([Runtime.InteropServices.Marshal]::SecureStringToBS
TR($input))
    }
    else
    {
        $input = Read-Host
    }

    [Console]::ResetColor()
    return $input
}
#-----Handle
input-----
"-----"
$acceptEULA = GetInput -prompt 'Accept Linux View Agent EULA in tar bundle ("yes" or "no")' -
IsPassword $false
if ($acceptEULA -ne "yes")
{
    write-host -ForegroundColor Red "You need accept the EULA with 'yes'(case sensitive)"
    exit
}
$svcAddress = GetInput -prompt "Your vCenter address" -IsPassword $false
$svcAdmin = GetInput -prompt "Your vCenter admin user name" -IsPassword $false
$svcPassword = GetInput -prompt "Your vCenter admin user password" -IsPassword $true
"-----"
$hostAdmin = GetInput -prompt 'Your ESXi host admin user name, such as root' -IsPassword $false
$hostPassword = GetInput -prompt "Your ESXi admin user password" -IsPassword $true
"-----"
$guestUser = GetInput -prompt 'Your VM guest OS user name' -IsPassword $false
$guestPassword = GetInput -prompt 'Your VM guest OS user password' -IsPassword $true
"-----"
$agentInstaller = GetInput -prompt 'Type the View Agent tar ball path' -IsPassword $false
"-----"
$UpgradeToManagedVM = GetInput -prompt 'Upgrade to managed VM ("yes" or "no")' -IsPassword $false
if (($UpgradeToManagedVM -ne "yes") -AND $UpgradeToManagedVM -ne "no")
```

```

{
    write-host -ForegroundColor Red "You need select 'yes' or 'no'(case sensitive)"
    exit
}
$installSmartcard = GetInput -prompt 'Install the Smartcard redirection feature ("yes" or "no")' -
IsPassword $false
if (($installSmartcard -ne "yes") -AND $installSmartcard -ne "no")
{
    write-host -ForegroundColor Red "You need select 'yes' or 'no'(case sensitive)"
    exit
}
"-----"

#$csvFile = Read-Host 'Csv File '
$csvFile = '.\CloneVMs.csv'

#check if file exists
if (!(Test-Path $agentInstaller))
{
    write-host -ForegroundColor Red "installer File not found"
    exit
}

#check if file exists
if (!(Test-Path $csvFile))
{
    write-host -ForegroundColor Red "CSV File not found"
    exit
}
#-----
Functions-----
function GetSourceInstallerMD5()
{
    $agentInstallerPath = Convert-Path $agentInstaller;
    $md5 = New-Object -TypeName System.Security.Cryptography.MD5CryptoServiceProvider;
    $md5HashWithFormat =
[System.BitConverter]::ToString($md5.ComputeHash([System.IO.File]::ReadAllBytes($agentInstallerPath)))
;
    $md5Hash = ($md5HashWithFormat.replace("-", "")).ToLower();
    return $md5Hash;
}

#-----
Main-----
#Get installer MD5Sum
$installerMD5Hash = GetSourceInstallerMD5;

#Connect to vCenter
$VC_Conn_State = Connect-VIServer $vcAddress -user $vcAdmin -password $vcPassword
if([string]::IsNullOrEmpty($VC_Conn_State))
{
    Write-Host 'Exit since failed to login vCenter'
    exit
}
else

```

```

{
    Write-Host 'vCenter is connected'
}

#Read input CSV file
$csvData = Import-CSV $csvFile

$destFolder = "/home/$guestUser/"

#Handle VMs one by one
foreach ($line in $csvData)
{
    "`n-----"
    $VMName = $line.VMName
    write-host -ForegroundColor Yellow "VM: $VMName`n"

    $cmd = "rm -rf VMware-*linux-*"
    Write-Host "Run cmd '$cmd' in VM '$VMName' with user '$guestUser'"
    Invoke-VMScript -HostUser $hostAdmin -HostPassword $hostPassword -VM $VMName -GuestUser
$guestUser -GuestPassword $guestPassword -Confirm:$false -ScriptType Bash -ScriptText $cmd

    #Upload installer tar ball to Linux VM
    Write-Host "Upload File '$agentInstaller' to '$destFolder' of VM '$VMName' with user '$guestUser'"
    Copy-VMGuestFile -HostUser $hostAdmin -HostPassword $hostPassword -VM $VMName -GuestUser
$guestUser -GuestPassword $guestPassword -Confirm:$false -LocalToGuest -Destination $destFolder -
Source $agentInstaller

    #Check the uploaded installer md5sum
    $cmd = "md5sum VMware-*linux-*"
    Write-Host "Run cmd '$cmd' in VM '$VMName' with user '$guestUser'"
    $output = Invoke-VMScript -HostUser $hostAdmin -HostPassword $hostPassword -VM $VMName -
GuestUser $guestUser -GuestPassword $guestPassword -Confirm:$false -ScriptType Bash -ScriptText $cmd

    if($output.Contains($installerMD5Hash))
    {
        Write-Host $VMName": Uploaded installer's MD5Sum matches the local installer's MD5Sum";
        Write-Host $VMName": Extract the installer and do installation";
        $cmd = "tar -xzf VMware-*linux-*.tar.gz"
        Write-Host "Run cmd '$cmd' in VM '$VMName' with user '$guestUser'"
        Invoke-VMScript -HostUser $hostAdmin -HostPassword $hostPassword -VM $VMName -GuestUser
$guestUser -GuestPassword $guestPassword -Confirm:$false -ScriptType Bash -ScriptText $cmd

        $cmd = "sudo setenforce 0";
        Write-Host "Set the selinux to permissive mode: $cmd"
        Invoke-VMScript -HostUser $hostAdmin -HostPassword $hostPassword -VM $VMName -GuestUser
$guestUser -GuestPassword $guestPassword -Confirm:$false -ScriptType Bash -ScriptText $cmd

        $cmd = "sudo killall /usr/lib/vmware/viewagent/VMwareBlastServer/VMwareBlastServer"
        Write-Host "Stop VMwareBlastServer before upgrading: $cmd"
        Invoke-VMScript -HostUser $hostAdmin -HostPassword $hostPassword -VM $VMName -GuestUser
$guestUser -GuestPassword $guestPassword -Confirm:$false -ScriptType Bash -ScriptText $cmd

        #Run the upgrade command.
        $cmd = "cd VMware-*linux-* && sudo ./install_viewagent.sh -A yes -m $installSmartcard -M
$UpgradeToManagedVM"

```

```

    Write-Host "Run upgrade cmd in VM '$VMName' with user '$guestUser': $cmd"
    Invoke-VMScript -HostUser $hostAdmin -HostPassword $hostPassword -VM $VMName -GuestUser
$guestUser -GuestPassword $guestPassword -Confirm:$false -ScriptType Bash -ScriptText $cmd

    $cmd = "sudo shutdown -r +1&"
    Write-Host "Reboot to apply the View Agent installation"
    Invoke-VMScript -HostUser $hostAdmin -HostPassword $hostPassword -VM $VMName -GuestUser
$guestUser -GuestPassword $guestPassword -Confirm:$false -ScriptType Bash -ScriptText $cmd
}
else
{
    Write-Host $VMName": Uploaded installer's MD5Sum does NOT match the local installer's MD5Sum";
    Write-Host $VMName": Skip the installation. Please check your network and VMware Tools
status";
    exit;
}
}
Disconnect-VIServer $vcAddress -Confirm:$false
exit

```

スクリプトの実行

このスクリプトを実行すると、次のメッセージが表示されます。

```

PowerCLI C:\scripts> .\InstallAgent.ps1

-----
Accept Linux Horizon Agent EULA in tar bundle ("yes" or "no"): yes
Your vCenter address: 10.117.44.17
Your vCenter admin user name: administrator
Your vCenter admin user password: *****

-----
Your ESXi host admin user name, such as root: root
Your ESXi host admin user password: *****

-----
Your VM guest OS user name: ViewUser
Your VM guest OS user password: *****

-----
Type the Horizon Agent tar ball path. Please take care of the installer arch: .\VMware-viewagent-
linux-x86_64-x.y.z-1234567.tar.gz

-----
Upgrade to managed VM ("yes" or "no"): yes
Install the Smartcard redirection feature ("yes" or "no"): no

```

SSH を使用して Linux 仮想マシンで Horizon Agent をアップグレードするサンプル スクリプト

次のサンプル スクリプトをカスタマイズして使用し、複数の Linux 仮想マシン (VM) で Horizon Agent をアップグレードできます。このスクリプトでは SSH を使用して、Linux 仮想マシンでコマンドを実行します。

改ページせずにスクリプトの内容をコピーして貼り付けるには、このトピックの HTML 版を使用します。HTML 版は https://www.vmware.com/support/pubs/view_pubs.html にある Horizon 7 のドキュメントのページから入手できます。

スクリプト入力

このスクリプトは、[Linux デスクトップを展開するサンプル PowerCLI スクリプトの入力ファイル](#)で説明しているように 1 つの入力ファイルを読み取ります。また、このスクリプトは、次の情報をインタラクティブに確認します。

- Horizon Agent EULA（エンド ユーザー使用許諾契約書）の承諾
- vCenter Server の IP アドレス
- vCenter Server 管理者のログイン名
- vCenter Server 管理者のパスワード
- ESXi ホスト管理者のログイン名
- ESXi ホスト管理者のパスワード
- Linux ゲスト OS のユーザー ログイン名
- Linux ゲスト OS のユーザー パスワード
- Horizon Agent の tar ボールのパス
- 管理対象の仮想マシンへアップグレード
- スマートカード リダイレクト機能をインストール

スクリプトのコンテンツ

```
<#
Upload the Linux Agent installer tar ball and re-install
#>

#-----
Functions-----
function GetInput
{
    Param($prompt, $IsPassword = $false)
    $prompt = $prompt + ": "
    Write-Host $prompt -NoNewLine
    [Console]::ForegroundColor = "Blue"
    if ($IsPassword)
    {
        $input = Read-Host -AsSecureString
        $input =
[Runtime.InteropServices.Marshal]::PtrToStringAuto([Runtime.InteropServices.Marshal]::SecureStringToBS
TR($input))
    }
    else
    {
        $input = Read-Host
    }

    [Console]::ResetColor()
    return $input
}
```

```

}
function Check_SSH_Client
{
    Param($IsPlink, $IsPSCP)
    if ($IsPlink)
    {
        if (Test-Path ".\plink.exe")
        {
            write-host -ForegroundColor Yellow 'SSH client "plink.exe" found'
        }
        else
        {
            write-host -ForegroundColor Red 'SSH client "plink.exe" not found, please download from
its official web site'
            exit
        }
    }
    if ($IsPSCP)
    {
        if (Test-Path ".\pscp.exe")
        {
            write-host -ForegroundColor Yellow 'SSH client "pscp.exe" found'
        }
        else
        {
            write-host -ForegroundColor Red 'SSH client "pscp.exe" not found, please download from its
official web site'
            exit
        }
    }
}

function RunCmdViaSSH
{
    Param($VM_Name, $User, $Password, $Cmd, $returnOutput = $false)

    $VM= Get-VM $VM_Name
    $IP = $VM.guest.IPAddress[0]
    write-host "Run cmd on $VM_Name ($IP)"
    if($returnOutput)
    {
        $command = "echo yes | .\plink.exe -ssh -l $user -pw $password $IP " + '"' + $cmd + '"'
        $output = Invoke-Expression $command
        return $output
    }
    else
    {
        echo yes | .\plink.exe -ssh -l $user -pw $password $IP "$cmd"
    }
}

function UploadFileViaSSH
{
    Param($VM_Name, $User, $Password, $LocalPath, $DestPath)

```

```

$VM= Get-VM $VM_Name
$IP = $VM.guest.IPAddress[0]
$command = "echo yes | .\pscp.exe -l $User -pw $Password $LocalPath $IP" + ":" + "$DestPath"
write-host "Upload file $LocalPath to VM $VM_Name with user $User"
Invoke-Expression $command
}

#-----Handle
input-----
"-----"
Check_SSH_Client -IsPlink $true -IsPSCP $true
"-----"
$acceptEULA = GetInput -prompt 'Accept Linux View Agent EULA in tar bundle ("yes" or "no")' -
IsPassword $false
if ($acceptEULA -ne "yes")
{
    write-host -ForegroundColor Red "You need accept the EULA with 'yes'(case sensitive)"
    exit
}
$svcAddress = GetInput -prompt "Your vCenter address" -IsPassword $false
$svcAdmin = GetInput -prompt "Your vCenter admin user name" -IsPassword $false
$svcPassword = GetInput -prompt "Your vCenter admin user password" -IsPassword $true
"-----"
$guestUser = GetInput -prompt 'Your VM guest OS user name' -IsPassword $false
$guestPassword = GetInput -prompt 'Your VM guest OS user password' -IsPassword $true
"-----"
$agentInstaller = GetInput -prompt 'Type the View Agent tar ball path' -IsPassword $false
"-----"
$UpgradeToManagedVM = GetInput -prompt 'Upgrade to managed VM ("yes" or "no")' -IsPassword $false
if (($UpgradeToManagedVM -ne "yes") -AND $UpgradeToManagedVM -ne "no")
{
    write-host -ForegroundColor Red "You need select 'yes' or 'no'(case sensitive)"
    exit
}
$installSmartcard = GetInput -prompt 'Install the Smartcard redirection feature ("yes" or "no")' -
IsPassword $false
if (($installSmartcard -ne "yes") -AND $installSmartcard -ne "no")
{
    write-host -ForegroundColor Red "You need select 'yes' or 'no'(case sensitive)"
    exit
}
"-----"

#$csvFile = Read-Host 'Csv File '
$csvFile = '.\CloneVMs.csv'

#check if file exists
if (!(Test-Path $agentInstaller))
{
    write-host -ForegroundColor Red "installer File not found"
    exit
}

#check if file exists

```

```

if (!(Test-Path $csvFile))
{
write-host -ForegroundColor Red "CSV File not found"
exit
}
#-----
Functions-----
function GetSourceInstallerMD5()
{
    $agentInstallerPath = Convert-Path $agentInstaller;
    $md5 = New-Object -TypeName System.Security.Cryptography.MD5CryptoServiceProvider;
    $md5HashWithFormat =
[System.BitConverter]::ToString($md5.ComputeHash([System.IO.File]::ReadAllBytes($agentInstallerPath)))
;
    $md5Hash = ($md5HashWithFormat.replace("-", "")).ToLower();
    return $md5Hash;
}

#-----
Main-----
#Get installer MD5Sum
$installerMD5Hash = GetSourceInstallerMD5;

#Connect to vCenter
$VC_Conn_State = Connect-VIServer $vcAddress -user $vcAdmin -password $vcPassword
if([string]::IsNullOrEmpty($VC_Conn_State))
{
    Write-Host 'Exit since failed to login vCenter'
    exit
}
else
{
    Write-Host 'vCenter is connected'
}

#Read input CSV file
$csvData = Import-CSV $csvFile

$destFolder = "/home/$guestUser/"

#Handle VMs one by one
foreach ($line in $csvData)
{
    "`n-----"
    $VMName = $line.VMName
    write-host -ForegroundColor Yellow "VM: $VMName`n"

    $cmd = "rm -rf VMware-*linux-*"
    Write-Host "Run cmd '$cmd' in VM '$VMName' with user '$guestUser'"
    RunCmdViaSSH -VM_Name $VMName -User $guestUser -Password $guestPassword -Cmd $cmd

    #Upload installer tar ball to Linux VM
    Write-Host "Upload File '$agentInstaller' to '$destFolder' of VM '$VMName' with user '$guestUser'"
    UploadFileViaSSH -VM_Name $VMName -User $guestUser -Password $guestPassword -LocalPath

```

```
$agentInstaller -DestPath $destFolder

#Check the uploaded installer md5sum
$cmd = "md5sum VMware-*linux-*"
Write-Host "Run cmd '$cmd' in VM '$VMName' with user '$guestUser'"
$output = RunCmdViaSSH -VM_Name $VMName -User $guestUser -Password $guestPassword -Cmd $cmd -
$returnOutput $true

if($output.Contains($installerMD5Hash))
{
    Write-Host $VMName": Uploaded installer's MD5Sum matches the local installer's MD5Sum";
    Write-Host $VMName": Extract the installer and do installation";

    $cmd = "tar -xzf VMware-*linux-*.tar.gz"
    Write-Host "Run cmd '$cmd' in VM '$VMName' with user '$guestUser'"
    RunCmdViaSSH -VM_Name $VMName -User $guestUser -Password $guestPassword -Cmd $cmd

    $cmd = "sudo setenforce 0";
    Write-Host "Set the selinux to permissive mode: $cmd"
    RunCmdViaSSH -VM_Name $VMName -User $guestUser -Password $guestPassword -Cmd $cmd

    $cmd = "sudo killall /usr/lib/vmware/viewagent/VMwareBlastServer/VMwareBlastServer"
    Write-Host "Stop VMwareBlastServer before upgrading: $cmd"
    RunCmdViaSSH -VM_Name $VMName -User $guestUser -Password $guestPassword -Cmd $cmd

    #Run the upgrade command.
    $cmd = "cd VMware-*linux-* && sudo ./install_viewagent.sh -r yes -A yes -m $installSmartcard
-M $UpgradeToManagedVM"
    Write-Host "Run upgrade cmd in VM '$VMName' with user '$guestUser': $cmd"
    RunCmdViaSSH -VM_Name $VMName -User $guestUser -Password $guestPassword -Cmd $cmd
    Write-Host -ForegroundColor Yellow "Linux Agent installer will reboot the Linux VM after
upgrade, and you may hit the ssh connection closed error message, which is expectation"
}
else
{
    Write-Host $VMName": Uploaded installer's MD5Sum does NOT match the local installer's MD5Sum";
    Write-Host $VMName": Skip the installation. Please check your network and VMware Tools
status";
    exit;
}
}
Disconnect-VIServer $vcAddress -Confirm:$false
exit
```

スクリプトの実行

このスクリプトを実行すると、次のメッセージが表示されます。

```
PowerCLI C:\scripts> .\InstallAgent.ps1
-----
Accept Linux Horizon Agent EULA in tar bundle ("yes" or "no"): yes
Your vCenter address: 10.117.44.17
Your vCenter admin user name: administrator
Your vCenter admin user password: *****
```

```

-----
Your VM guest OS user name: ViewUser
Your VM guest OS user password: *****
-----

Type the Horizon Agent tar ball path. Please take care of the installer arch: .\VMware-viewagent-
linux-x86_64-x.y.z-1234567.tar.gz
-----

----
Upgrade to managed VM ("yes" or "no"): yes
Install the Smartcard redirection feature ("yes" or "no"): no

```

Linux 仮想マシンで操作を実行するサンプル スクリプト

次のサンプル スクリプトをカスタマイズして使用し、複数の Linux 仮想マシン (VM) で操作を実行できます。操作には、仮想マシンのパワーオン、パワーオフ、シャットダウン、および削除が含まれます。

このスクリプトによって、vCenter Server から仮想マシンを削除できますが、View から削除できません。

改ページせずにスクリプトの内容をコピーして貼り付けるには、このトピックの HTML 版を使用します。HTML 版は https://www.vmware.com/support/pubs/view_pubs.html にある Horizon 7 のドキュメントのページから入手できます。

スクリプト入力

このスクリプトは、[Linux デスクトップを展開するサンプル PowerCLI スクリプトの入力ファイル](#)で説明しているように 1 つの入力ファイルを読み取ります。また、このスクリプトは、次の情報をインタラクティブに確認します。

- vCenter Server の IP アドレス
- vCenter Server 管理者のログイン名
- vCenter Server 管理者のパスワード
- 実行するアクション。パワーオン、パワーオフ、ゲストのシャットダウン、仮想マシンの再起動、仮想マシン ゲストの再起動、仮想マシンの削除のいずれかです。
- 仮想マシンでの操作間の待機時間（秒単位）

スクリプトのコンテンツ

```

<#
.DESCRIPTION
The Tool supports:
1. Power off VMs
2. Power on VMs
3. Shutdown VMs
4. Restart VMs
5. Restart VM guest
6. Delete VMs from Disk
.NOTES
#>

#----- Functions -----

```

```

function GetInput
{
    Param($prompt, $IsPassword = $false)
    $prompt = $prompt + ": "
    Write-Host $prompt -NoNewLine
    [Console]::ForegroundColor = "Blue"
    if ($IsPassword)
    {
        $input = Read-Host -AsSecureString
        $input =
[Runtime.InteropServices.Marshal]::PtrToStringAuto([Runtime.InteropServices.Marshal]::SecureStringToBS
TR($input))
    }
    else
    {
        $input = Read-Host
    }

    [Console]::ResetColor()
    return $input
}

function IsVMExists ($VMExists)
{
    Write-Host "Checking if the VM $VMExists Exists"
    [bool]$Exists = $false

    #Get all VMS and check if the VMs is already present in VC
    $listvm = Get-vm
    foreach ($lvm in $listvm)
    {
        if($VMExists -eq $lvm.Name )
        {
            $Exists = $true
            Write-Host "$VMExists is Exist"
        }
    }
    return $Exists
}

function Delete_VM($VMToDelete)
{
    Write-Host "Deleting VM $VMToDelete"
    Get-VM $VMToDelete | where { $_.PowerState -eq "PoweredOn" } | Stop-VM -confirm:$false
    Get-VM $VMToDelete | Remove-VM -DeleteFromDisk -confirm:$false
}

#----- Handle input -----
"-----"
$vcAddress = GetInput -prompt "Your vCenter address" -IsPassword $false
$vcAdmin = GetInput -prompt "Your vCenter admin user name" -IsPassword $false
$vcPassword = GetInput -prompt "Your vCenter admin user password" -IsPassword $true
"-----"
$action = GetInput -prompt 'Select action: 1). Power On 2). Power Off 3) Shutdown VM Guest 4).

```

```

Restart VM 5). Restart VM Guest 6). Delete VM' -IsPassword $false
$sleepTime = GetInput -prompt 'Wait time (seconds) between each VM' -IsPassword $false
"-----"
[Console]::ForegroundColor = "Yellow"
switch ($action)
{
    1
    {
        "Your selection is 1). Power On"
    }
    2
    {
        "Your selection is 2). Power Off"
    }
    3
    {
        "Your selection is 3) Shutdown"
    }
    4
    {
        "Your selection is 4). Restart VM"
    }
    5
    {
        "Your selection is 5). Restart VM Guest"
    }
    6
    {
        "Your selection is 6). Delete VM"
    }
    default
    {
        "Invalid selection for action: $action"
        exit
    }
}
[Console]::ResetColor()
$csvFile = '.\CloneVMs.csv'

#check if file exists
if (!(Test-Path $csvFile))
{
    write-host -ForegroundColor Red "CSV File not found"
    exit
}
"-----"

#----- Main -----
#Read input CSV file
Disconnect-VIServer $vcAddress -Confirm:$false
#Connect-VIServer $vcAddress -ErrorAction Stop -user $vcAdmin -password $vcPassword
Connect-VIServer $vcAddress -user $vcAdmin -password $vcPassword
$csvData = Import-CSV $csvFile

foreach ($line in $csvData)

```



```
{
    $VMName = $line.VMName
    switch ($action)
    {
        1
        {
            Get-VM $VMName | Start-VM -Confirm:$false
        }
        2
        {
            Get-VM $VMName | Stop-VM -Confirm:$false
        }
        3
        {
            Get-VM $VMName | Shutdown-VMGuest -Confirm:$false
        }
        4
        {
            Get-VM $VMName | Restart-VM -Confirm:$false
        }
        5
        {
            Get-VM $VMName | Restart-VMGuest -Confirm:$false
        }
        6
        {
            if (IsVMExists ($VMName))
            {
                Delete_VM ($VMName)
            }
        }
        default{}
    }
    Start-Sleep -s $sleepTime
}

Disconnect-VIServer $vcAddress -Confirm:$false
exit
```

スクリプトの実行

このスクリプトを実行すると、次のメッセージが表示されます。

```
PowerCLI C:\scripts> .\VMOperations.ps1
Your vCenter address: 10.117.44.17
Your vCenter admin user name: administrator
Your vCenter admin user password: *****

-----
Select action: 1). Power On 2). Power Off 3) Shutdown VM Guest 4). Restart VM 5). Restart VM Guest
6). Delete VM: 1
Wait time (seconds) between each VM: 20
-----

Your selection is 6). Delete VM
```

パワーオン、仮想マシンの再起動、および仮想マシン ゲストの再起動の操作については、仮想マシン間における待機時間に少なくとも 20 秒を指定して、操作が失敗する原因となることがあるブート ストームの状況を回避してください。

Linux デスクトップのトラブルシューティング

9

Linux デスクトップの管理時に特定の問題が発生する可能性があります。問題を診断および解決するためにさまざまな手順を実行できます。

この章には、次のトピックが含まれています。

- [Horizon 7 for Linux マシンの診断情報の収集](#)
- [リモート デスクトップとクライアント ホストの間でのコピーと貼り付けに関するトラブルシューティング](#)
- [TCP 接続の受信を許可するための Linux ファイアウォールの構成](#)
- [View Agent が iPad Pro Horizon Client で切断できない](#)
- [ドラッグ アンド ドロップ後、SLES 12 SP1 デスクトップが自動更新しない](#)
- [SSO がパワーオフ エージェントに接続できない](#)
- [Linux 版手動デスクトップ プール作成後の接続不能な仮想マシン](#)

Horizon 7 for Linux マシンの診断情報の収集

VMware のテクニカル サポートが Horizon 7 for Linux マシンの問題を診断して解決する際に役立つ診断情報を収集できます。マシンの構成情報を収集して圧縮した tar ボールに記録するデータ収集ツール (DCT) バンドルを作成します。

手順

- 1 必要な権限を持つユーザーとして Linux 仮想マシンにログインします。
- 2 コマンド プロンプトを開いて、`dct-debug.sh` スクリプトを実行します。

```
sudo /usr/lib/vmware/viewagent/bin/dct-debug.sh
```

スクリプトによって、DCT バンドルを含む tar ボールが生成されます。例：

```
ubuntu-12-vdm-sdct-20150201-0606-agent.tgz
```

tar ボールは、スクリプトが実行されたディレクトリ（現在の作業ディレクトリ）に生成されます。

リモート デスクトップとクライアント ホストの間でのコピーと貼り付けに関するトラブルシューティング

リモート デスクトップとクライアント ホストの間でのコピーと貼り付けは、最大 1 MB のデータでサポートされ、処理に 3 秒以上かかります。この問題は、サイズの小さなデータのコピーと貼り付けを行う場合には発生しません。

問題

SLED 11 SP3/SP4 デスクトップ向けに 1 つの vCPU と 1 GB のメモリを構成するとき、リモート デスクトップとローカル クライアント ホストの間でのコピーと貼り付けに 3 秒以上かかることがあります。

原因

コピーと貼り付けの遅延は、SLED 11 SP3/SP4 のオペレーティング システム API が古いために発生することがあります。

解決方法

- ◆ SLED 11 SP3/SP4 に 2 つの vCPU と 2 GB のメモリを構成します。

TCP 接続の受信を許可するための Linux ファイアウォールの構成

ユーザーが自分の Linux デスクトップに接続できるようにするには、Horizon Client デバイス、セキュリティ サーバ、および View 接続サーバから受信する TCP 接続をデスクトップが受け入れる必要があります。

Ubuntu および Kylin ディストリビューションでは、iptables ファイアウォールがデフォルトで構成されており、入力ポリシーが ACCEPT に設定されています。

RHEL および CentOS ディストリビューションでは、可能な場合、Horizon Agent インストーラ スクリプトが、入力ポリシーを ACCEPT にして iptables ファイアウォールを構成します。

RHEL や CentOS ゲスト OS の iptables では、Blast ポート 22443 からの新しい接続について入力ポリシーが ACCEPT になっていることを確認します。

BSG が有効な場合、クライアント接続はセキュリティ サーバまたは View 接続サーバの BSG を介して Horizon Client デバイスから Linux デスクトップに送られます。BSG が有効ではない場合、Horizon Client デバイスは Linux デスクトップに直接接続されます。

View Agent が iPad Pro Horizon Client で切断できない

iPad Pro Horizon Client で再始動またはシャットダウンした後に、SUSE View Agent の接続が切断できません。

問題

iPad Pro Horizon Client で、SUSE 仮想マシンを再起動またはシャットダウンする際、デスクトップは応答しません。View Agent は切断できません。

原因

再起動またはシャットダウンの操作の後に、SUSE マシンは Horizon Client にメッセージを正確に送信していない場合があります。

解決方法

- ◆ iPad Pro Horizon Client からデスクトップ接続を手動で切断してください。

ドラッグ アンド ドロップ後、SLES 12 SP1 デスクトップが自動更新しない

gnome ターミナルをドラッグ アンド ドロップする場合、SLES 12 SP1 はマルチモニタ モードで自動更新しません。

問題

マルチモニタ モードで SLES 12 SP1 を起動し、ウィンドウ モードに戻り、gnome ターミナルをドラッグ アンド ドロップする場合は、デスクトップは自動的に更新しません。

原因

gnome ターミナルは、ドラッグ アンド ドロップ操作に応答しません。

解決方法

- 1 gnome シェルを終了します。



```
kill -9 `pidof gnome-shell`
```

- 2 gnome シェルを再起動します。

SSO がパワーオフ エージェントに接続できない

SSO がパワーオフ エージェントに接続しません。

問題

ブローカーとしてログインし、エージェントに接続する場合、SSO はパワーオフエージェントに接続しません。

解決方法

- ◆ 手動でデスクトップへログインするか切断します。次に、エージェントに再接続します。

Linux 版手動デスクトップ プール作成後の接続不能な仮想マシン

仮想マシン状態が応答状態にない。

問題

手動デスクトップ プールの作成後、仮想マシン ステータスは [エージェントの待機]、または [接続不能] になっている可能性があります。

原因

仮想マシン状態が [エージェントの待機] または [接続不能] になる場合には、いくつかのユーザー エラー構成またはセットアップに関する原因が考えられます。

- オプション `machine.id` が仮想マシンの `vmx` 構成ファイルに存在していることを確認します。

そのオプションが存在していない場合、仮想マシンがデスクトップ プールに正しく追加されていることを確認します。存在している場合は、デスクトップ プールを再作成し、ブローカで `vmx` 構成ファイルにそのオプションを再度書き込みます。

- VMware Tools または Open VM Tools が正しくインストールされていることを確認します。

VMware Tools または Open VM Tools をインストールするステップが正しく実行されていなかった場合、`vmware-rpctool` コマンドは Linux 仮想マシンの `PATH` に存在しない可能性があります。ガイドに従って VMware Tools または Open VM Tools をインストールする必要があります。

インストールの終了後、以下のコマンドを実行します。

```
#vmware-rpctool "machine.id.get"
```

`machine.id` の値は仮想マシンの `vmx` 構成ファイルから一覧表示されます。

- ブローカの完全修飾ドメイン名 (FQDN) をエージェント Linux 仮想マシンで IP アドレスに名前解決できるかどうかを確認します。