Lpi.305-300.v2026-06-16.q86

試験コード：	305-300
試験名称：	LPIC-3 Exam 305: Virtualization and Containerization
認証ベンダー：	Lpi
無料問題の数：	86
バージョン：	v2026-06-16
ページの閲覧量：	106
問題集の閲覧量：	900

https://www.jpnshiken.com/shiken/Lpi.305-300.v2026-06-16.q86.html

質問: **1**
xlコマンドのどのサブコマンドを使用すると、現在Xenゲストドメインに接続されているブロックデバイスを一覧表示できますか？
正解:
ブロックリスト
Explanation:
Xen仮想化環境では、xltoolstackはゲストドメインの管理と検査を行うためのコマンドを提供します。Xenのドキュメントによると、block-listサブコマンドは、特定のXenゲストドメインに現在接続されているすべてのブロックデバイスを一覧表示するために使用されます。
このコマンドは、仮想デバイス名、バックエンドデバイス、アクセスモードなどの情報を表示し、管理者がディスクの接続状況を確認したり、ストレージ関連の問題をトラブルシューティングしたりできるようにします。これは、ブロック接続またはブロック切断を使用してディスクを接続または切断した後によく使用されます。
したがって、正しく文書化されているサブコマンドはblock-listです。

質問: **2**
Dockerfileに以下の行が含まれている場合：
作業ディレクトリ /
実行 cd /tmp
RUN echo test > test
ファイルtestはどこにありますか？
A. コンテナイメージ内のテスト。
B. コンテナイメージ内の /root/tesc。
C. Dockerfile が格納されているディレクトリでテストを実行します。
D. コンテナイメージ内の /ting/test。
E. docker build を実行しているシステムの /tmp/test。
正解: **A (**コメントを発表する**)**

質問: **3**

以下のコマンドのうち、ハードウェア仮想化拡張機能を使用してQEMU仮想マシンを起動するのはどれですか？
A. qvm start -vmx -drive file=debian.img -cdrom debian.iso -m 1024 -boot d
B. vm -kvm -drive file=debian.img -cdrom debian.iso -m 1024 -boot d
C. qvirt -create -drive file=debian.img -cdrom debian.iso -m 1024 -boot d -driver hvm
D. qemu -accel kvm -drive file-debian.img -cdrom debian.iso -m 1024 -boot d
E. qemu-hw -create -drive file=debian.img -cdrom debian.iso -m 1024 -boot d
正解: **(**正解を表示します**)**

質問: **4**
Linuxベースの仮想化環境で一般的に使用されるディスクイメージ形式はどれですか？（該当するものをすべて選択してください）
A. RAW
B. VMDK
C. QCOW2
D. VHD
正解: **A,B,C,D (**コメントを発表する**)**
Linuxベースの仮想化環境は、複数のハイパーバイザーやクラウドプラットフォームとの互換性を確保するために、幅広いディスクイメージ形式をサポートしています。仮想化に関するドキュメントによると、RAW、VMDK、QCOW2、VHDはすべて一般的に使用される形式です。
RAW画像は、最小限のオーバーヘッドで最大限のパフォーマンスを発揮する、シンプルで構造化されていないディスクファイルです。
QCOW2（QEMU Copy-On-Write バージョン 2）は、スナップショット、シンプロビジョニング、圧縮、暗号化などの高度な機能をサポートしているため、KVM 環境で最も広く使用されているフォーマットです。VMDK は VMware 製品のネイティブディスクフォーマットですが、相互運用性や移行の目的で Linux 環境でも頻繁に使用されています。VH は一般的に Microsoft Hyper-V に関連付けられていますが、QEMU やクラウドプラットフォームでもサポートされています。
仮想化に関する注記では、QEMUやlibvirtといった最新のLinux仮想化ツールは、複数のディスクフォーマットに対応するように設計されていることが強調されています。この柔軟性により、管理者は仮想マシンを再構築することなく、異なるハイパーバイザーやクラウドプロバイダー間でワークロードを移行できます。
したがって、記載されているすべてのディスクイメージ形式は有効であり、Linuxベースの仮想化環境で一般的にサポートされています。

質問: **5**
libvirtドメイン定義の<domain>要素のtype属性において有効な値は次のうちどれですか？（2つ選択してください。）
A. cgroup
B. プロセス
C. 名前空間

D. イクス
E. kvm
正解: **D,E** (コメントを発表する)

質問: **6**
libvirtドメイン定義内の<os>要素のfirmware属性で有効な値は次のうちどれですか？ 2つ選択してください。）
A. SCSI
B. virtio
C. efi
D. バイオ
E. PCIe
正解: (正解を表示します)
説明
libvirtドメイン定義の<os>要素のファームウェア属性は、仮想マシンの起動に使用されるファームウェアの種類を指定します。この属性の有効な値はefiとbiosで、それぞれ拡張ファームウェアインターフェイス（EFI）と基本入出力システム（BIOS）に対応します。EFIはBIOSよりも多くの機能とセキュリティをサポートする新しい標準規格で、セキュアブートや高速な起動時間などが含まれます。BIOSは古い標準規格ですが、ほとんどのハイパーバイザーとオペレーティングシステムで広く互換性がありサポートされています。その他の値であるscsi、virtio、pcieはファームウェアとは関係なく、仮想マシンで使用できるさまざまな種類のデバイスまたはドライバに関連しています。参考文献：1（ファームウェア列挙型を検索）

質問: **7**
Packerはシステムイメージとどのように連携するのですか？
A. Packerはターゲットイメージ内にインストールする必要があり、準備タスクを実行するためにイメージの最初の起動時に実行されます。
B. Packer はイメージ内にクライアントをインストールします。このクライアントは、Packer サーバーから最新のテンプレートを取得してローカルに適用するために、cron を介して定期的に実行する必要があります。
C. Packer は、実行中のすべての Packer イメージの Packer デーモンにネットワーク経由で定期的に接続し、実行中のインスタンスにテンプレート全体を再適用します。
D. Packerはイメージをダウンロードして解凍し、イメージのファイルシステムに変更を加え、変更されたイメージを再パックして再度アップロードします。
E. Packerはソースイメージに基づいてインスタンスを作成し、ネットワーク接続を介してインスタンスを準備し、結果として得られたインスタンスを新しいシステムイメージとしてバンドルします。
正解: (正解を表示します)
Packerは、単一のソース構成から複数のプラットフォーム向けに同一のマシンイメージを自動的に作成するツールです。Packerは、ソースイメージ（開始点となる既存のイメージ）に基づいてインスタンスを作成します。次に、SSHやWinRMなどのネットワーク接続を介してインスタンスに接続し、インスタンス内でソフトウェアをインストールおよび構成するためのさまざまなコマンドとスクリプトを実行します。

その後、Packerはインスタンスをシャットダウンし、そこから新しいシステムイメージを作成します。この新しいシステムイメージは、新しいインスタンスの起動に使用できます。Packer
は、AWS、Azure、VMware、Dockerなど、多くのプラットフォームをサポートしています。
Packer は、ターゲットイメージ内にソフトウェアをインストールしたり、デーモンを実行したりしません。また、実行中のインスタンスに定期的に接続してテンプレートを再適用することもありません。Packer はソースイメージを直接変更するのではなく、変更されたインスタンスから新しいイメージを作成します。参考資料:
* Packer by HashiCorp
* HashiCorp Packer - 自動化されたマシンイメージの構築
* 概要 | Packer | HashiCorp開発者

質問: **8**
libvirtは設定データを保存するためにどのファイル形式を使用しますか？
A. INI形式のテキストファイル
B. SQLiteデータベース
C. XMLファイル
D. Javaライクなプロパティファイル
E. キーと値のペアを含むテキストファイル
正解: **(**正解を表示します**)**
Libvirtは、ドメイン、ネットワーク、ストレージなど、libvirt API内のオブジェクトの設定データをXMLファイルに保存します。これにより、将来のリリースでの拡張が容易になり、使用前にドキュメントの検証を行うことができます。
Libvirt は、質問に記載されている他のファイル形式は使用しません。参考文献:
* libvirt: XML形式
* LPIC-3 仮想化とコンテナ化 :トピック 305.1：仮想化の概念と理論

質問: **9**
次のツールのうち、VMのプロビジョニングやイメージ作成に通常使用されないものはどれですか？
A. Kubernetes
B. 放浪者
C. パッカー
D. cloud-init
正解: **(**正解を表示します**)**
VMプロビジョニングツールとイメージ作成ツールは、仮想マシンイメージの構築、構成、および初期化に使用されます。仮想化に関するドキュメントによると、Packerは自動イメージ作成専用に設計されており、Vagrantは再現可能な開発環境のプロビジョニングに使用され、cloud-inは初回起動時にインスタンスを構成するために使用されます。
しかし、Kubernetesはコンテナオーケストレーションプラットフォームであり、仮想マシンのプロビジョニングやイメージ作成ツールではありません。
インフラストラクチャが既にプロビジョニングされた後、コンテナ化されたワークロードを管理します。

ドキュメントでは、インフラストラクチャプロビジョニングツールとアプリケーションオーケストレーションプラットフォームが明確に区別されています。
そのため、Kubernetesは通常、仮想マシンのプロビジョニングやイメージの作成には使用されません。
したがって、正解はAです。

質問: **10**
空欄を埋める
virshopコマンドのどのサブコマンドを使用すると、仮想ネットワークのXML設定をエディタで展開して、その設定を変更できますか？（パラメータは指定せず、サブコマンドのみを指定してください。）
正解:
ネット編集

質問: **11**
完全仮想化に関して、以下の記述のうち正しいものはどれですか？（正しいものを2つ選択してください）
A. 完全な仮想化には常に追加のソフトウェアコンポーネントが必要であり、Linux だけでは実現できません。
B. 完全な仮想化は、より優れたパフォーマンスを提供する特別なCPU拡張機能によってサポートされる場合があります。
C. 完全仮想化では、ゲストオペレーティングシステムに変更を加える必要はありません。
D. 完全仮想化は、同じマシン上の非仮想化ベアメタルインストールと比較して、パフォーマンスに影響を与えません。
E. 完全仮想化はパフォーマンスに深刻な影響を与えるため、本番環境では使用しないでください。
正解: **B,C (**コメントを発表する**)**
完全仮想化とは、ハイパーバイザが基盤となるハードウェアの完全なシミュレーションを提供し、変更されていないゲストオペレーティングシステムが仮想化を意識することなく実行できるようにする仮想化手法です。仮想化に関するドキュメントでは、完全仮想化の決定的な特徴の1つとして、ゲストオペレーティングシステムを変更する必要がないことが一貫して述べられており、したがって選択肢Cが正解です。
さらに、最新の完全仮想化プラットフォームは、Intel VT-xやAMD-V CPU拡張機能などのハードウェア支援型仮想化の恩恵を受けています。これらの拡張機能は、仮想化タスクをプロセッサにオフロードすることでパフォーマンスを大幅に向上させるため、選択肢Bが正解です。
選択肢Aは、KVMのようなLinuxベースのハイパーバイザーがLinuxエコシステム内で完全な仮想化を実行できるため誤りです。選択肢Dは、仮想化はたとえ最小限であっても何らかのパフォーマンスオーバーヘッドを伴うため誤りです。選択肢Eは、エンタープライズデータセンターやクラウドプラットフォームなど、本番環境で完全な仮想化が広く使用されているため誤りです。
したがって、仮想化の原理とドキュメントに基づくと、正解はBとCです。

質問: **12**
KubernetesのPodに関して、以下の記述のうち正しいものはどれですか？（2つ選択してください。）

A. Pod のすべてのコンテナは同じノードで実行されます。
B. Podは常に自動的に作成され、明示的に設定することはできません。
C. Pod は、Kubernetes が実行できるワークロードの最小単位です。
D. Pod が失敗した場合、Kubernetes はデフォルトで別のノードで Pod を再起動します。
E. systemd は、Kubernetes ノード上の個々の Pod を管理するために使用されます。
正解: **A,C (**コメントを発表する**)**
Kubernetes の Pod は、同じネットワークおよびストレージ リソースを共有する 1 つ以上のコンテナの集合であり、コンテナの実行方法に関する仕様です。Pod は Kubernetes が実行できるワークロードの最小単位であり、より小さな単位に分割することはできません。したがって、オプション C が正解です。Pod のすべてのコンテナは同じノードで実行されます。ノードは Kubernetes のコンピューティング ハードウェアの最小単位です。ノードは、1 つ以上の Pod をホストする物理マシンまたは仮想マシンです。したがって、オプション A も正解です。Pod は常に自動的に作成されるわけではなく、明示的に構成することもできません。Pod は、YAML ファイルまたは JSON ファイルを使用するか、kubectl run や kubectl create などのコマンドを使用して手動で作成できます。Pod は、Deployment、ReplicaSet、StatefulSet などの上位レベルのコントローラーによって自動的に作成することもできます。したがって、オプション B は不正解です。Pod が失敗した場合、Kubernetes はデフォルトでは別のノードで Pod を再起動しません。Pod は本質的に一時的なものであり、いつでも終了または削除できます。Podがコントローラーによって管理されている場合、コントローラーは障害が発生したPodを置き換える新しいPodを作成しますが、それは同じノード上にあるとは限りません。
したがって、選択肢Dは誤りです。systemdはKubernetesノード上の個々のPodを管理するために使用されるものではありません。
systemdは、Linuxオペレーティングシステム用のシステムおよびサービスマネージャであり、dockerやkubeletなどのサービスを開始および停止できます。ただし、systemdはPodと直接やり取りしません。PodはkubeletサービスによってPodが管理されます。kubeletは各ノードで実行され、Kubernetesコントロールプレーンと通信するエージェントです。
したがって、選択肢Eは誤りです。参考文献：
* ポッド | Kubernetes
* Kubernetesポッドとは何ですか？ - Red Hat
* ポッド、クラスター、コンテナの違いは何ですか？
* Kubernetes Podとは？ | VMware用語集
* Kubernetesのノード、ポッド、クラスター：主な違い - CloudZero

質問: **13**
VagrantとDockerの両方で提供される機能はどれですか？ 3つ選択してください。）
A. どちらもホストファイルシステムからゲストにディレクトリを共有できます。
B. デフォルトでは、どちらもシステムイメージを仮想マシンではなくコンテナとして起動します。
C. どちらも必要なベースイメージをダウンロードできます。
D. どちらもベース画像に変更を加えることができます。
E. デフォルトでは、両方ともコンテナではなく仮想マシンとしてシステムイメージを起動します。

正解: **A,C,D** (コメントを発表する)
VagrantとDockerはどちらも、ホストのファイルシステムからゲストにディレクトリを共有できます。これにより、ゲストはホストのファイルやフォルダをコピーせずにアクセスできます。Vagrantはconfig.vmを使用します。
Vagrantfile の synced_folder オプションで共有フォルダを指定します。1. Docker は、docker run コマンドの -v または -- volume フラグを使用して、ホスト ディレクトリをコンテナ内のデータ ボリュームとしてマウントします。2.
VagrantとDockerはどちらも、必要なベースイメージをダウンロードできます。ベースイメージは、ゲスト環境を作成するための出発点となります。Vagrantは、Vagrantfileのconfig.vm.boxオプションを使用して、使用するベースイメージを指定します1。Dockerは、DockerfileのFROM命令を使用して、使用するベースイメージを指定します2。VagrantとDockerはどちらも、パブリックリポジトリまたはローカルソースからベースイメージをダウンロードできます。
VagrantとDockerはどちらも、ベースイメージに変更を加えることができます。変更とは、ゲスト環境をカスタマイズするためにベースイメージに加えられる修正や追加です。Vagrantはプロビジョナーを使用して、ゲストが起動した後にスクリプトやコマンドを実行します1。DockerはDockerfileの指示を使用して、ベースイメージ上でコマンドを実行し、新しいイメージを作成します2。VagrantとDockerはどちらも、変更を新しいイメージに保存したり、ゲストが破棄された後に変更を破棄したりできます。
* VagrantとDockerは、システムイメージの起動方法が異なります。Vagrantはデフォルトで、VirtualBox、VMware、Hyper-V1などのプロバイダを使用して、システムイメージを仮想マシンとして起動します。Dockerはデフォルトで、macOS、Linux、Windowsのネイティブコンテナ化機能を使用して、システムイメージをコンテナとして起動します。コンテナは一般的に仮想マシンよりも軽量で高速ですが、セキュリティと柔軟性は劣ります。参考資料 :1 :Vagrant vs. Docker | Vagrant | HashiCorp Developer 2 :Vagrant vs Docker :どちらが自分に合っているか？（両方でも可）- Kinsta Web Development Tools

質問: **14**
.dockerignoreファイルの目的は何ですか？
A. 派生イメージを構築する際に除外すべき、Dockerイメージ内に存在するファイルを一覧表示します。
B. Dockerイメージをビルドする際に、DockerがDockerデーモンに送信しないファイルを指定します。
C. Dockerによって提供されるボリュームとポートを無視する必要があるコンテナのルートファイルシステムに存在します。
D. Dockerがコンテナに自動的にアタッチしてはならないボリュームの最上位ディレクトリに配置する必要があります。
E. Dockerイメージをビルドする際に、Dockerfileのどの部分を無視するかを指定します。
正解: **B** (コメントを発表する)
説明
.dockerignore ファイルの目的は、Docker イメージをビルドする際に Docker が Docker デーモンに送信しないファイルを指定することです。.dockerignore ファイルは、ビルド コンテキスト（Dockerfile で使用できるファイルとフォルダーのセット）から除外するファイルまたはディレクトリのリストを含むテキスト ファイルです。.dockerignore ファイルを使用することで、サイズが大きいファイルや機密情報を含む

ファイル、または Docker イメージに関係のないファイルやディレクトリをデーモンに送信することを回避でき、ビルド プロセスの効率とセキュリティを向上させることができます。他のオプションは、.dockerignore ファイルの機能を説明していないため、正しくありません。
.dockerignore ファイル。オプション A は、.dockerignore ファイルが Docker イメージに存在するファイルには影響せず、ビルド中にデーモンに送信されるファイルのみに影響するため、誤りです。オプション C は、.dockerignore ファイルがコンテナのルート ファイルシステムには存在せず、Dockerfile と同じディレクトリに存在するため、誤りです。オプション D は、.dockerignore ファイルが Docker がコンテナにアタッチするボリュームには影響せず、ビルド コンテキストに含まれるファイルのみに影響するため、誤りです。オプション E は、.dockerignore ファイルが実行される Dockerfile の部分には影響せず、Dockerfile で使用可能なファイルのみに影響するため、誤りです。参照:
* .dockerignoreファイルとは何ですか？また、なぜ使用すべきなのでしょうか？
* Dockerfile リファレンス | Docker ドキュメント
* .dockerignore の使い方とその重要性 - Shisho Cloud

質問: **15**
空欄を埋める
新しいLinuxネームスペースでプロセスを実行するには、どのコマンドを使用しますか？（パスやパラメータは含めずに、コマンドのみを指定してください。）
正解:
共有を解除する

質問: **16**
Vagrantのデフォルトプロバイダは何ですか？
A. lxc
B. ハイパーV
C. バーチャルボックス
D. VMware Workstation
E. ドッカー
正解: **(**正解を表示します**)**
説明
Vagrant は、軽量で再現性があり、移植可能な開発環境を作成および構成できるツールです。Vagrant は複数のプロバイダーをサポートしており、プロバイダーは Vagrant が仮想マシンの作成と管理に使用するバックエンドです。デフォルトでは、VirtualBox が Vagrant のデフォルトプロバイダーです。VirtualBox は、無料、クロスプラットフォームであり、Vagrant によって長年サポートされているため、Vagrant を使用する上で最もアクセスしやすいプラットフォームです。VirtualBox がデフォルトプロバイダーであるため、新規ユーザーが Vagrant を使い始める際の障壁が最も低くなっています。ただし、ユーザーは好みやニーズに応じて、VMware、Hyper-V、Docker、LXC などの他のプロバイダーを使用することもできます。別のプロバイダーを使用するには、Vagrant プラグインとしてインストールし、Vagrant コマンドを実行する際に指定する必要があります。また、VAGRANT_DEFAULT_PROVIDER 環境変数を設定することで、デフォルトプロバイダーを変更することもできます。参考資料:

* デフォルトプロバイダー - プロバイダー | Vagrant | HashiCorp Developer1
* プロバイダー | Vagrant | HashiCorp Developer2
* VagrantのデフォルトプロバイダをVirtualbox3に設定する方法

有効的な**305-300**問題集はJPNTest.com提供され、**305-300**試験に合格することに役に立ちます！JPNTest.comは今最新**305-300**試験問題集を提供します。JPNTest.com 305-300試験問題集はもう更新されました。ここで**305-300**問題集のテストエンジンを手に入れます。最新版のアクセス、https://www.jpntest.com/shiken/305-300-mondaishu　**125**問、３０%ディスカウント、特別な割引コード: **JPNshiken**」

質問: **17**
Xenは、以下のゲストシステムのうち、どれをサポートしていますか？ 2つ選択してください。）
A. 外国人建築家ゲスト FA）
B. 準仮想化クエスト (PVI
C. エミュレートされたゲスト
D. コンテナ仮想化ゲスト
E. 完全仮想化ゲスト
正解: (正解を表示します)
Xenは、準仮想化ゲスト PV）と完全仮想化ゲスト HVM）の2種類のゲストシステムをサポートしています。
準仮想化ゲスト PV）は、Xenハイパーバイザ上で動作するように変更されたゲストです。PVゲストは、ハイパーコールを介してハイパーバイザと通信する特別なカーネルを使用し、I/Oデバイスには準仮想化ドライバを使用します。PVゲストはHVMゲストよりも高速かつ効率的に動作しますが、ゲストオペレーティングシステムがXenに移植され、Xen ABI12をサポートしている必要があります。
* 完全仮想化ゲスト (HVM) は、Xen ハイパーバイザ上で変更されていないオペレーティングシステムを実行するゲストです。HVM は、Intel VT-x や AMD-V などのハードウェア仮想化拡張機能を使用して、ゲスト用の仮想プラットフォームを作成します。HVM ゲストは、ハードウェア アーキテクチャをサポートする任意のオペレーティングシステムを実行できますが、PV ゲストよりもオーバーヘッドとパフォーマンスの低下が大きくなります。HVM ゲストは、I/O デバイスに準仮想化ドライバを使用してパフォーマンスを向上させることもできます 12。
他の選択肢は正しくありません。Xenは、外部アーキテクチャゲスト FA）、エミュレートゲスト、またはコンテナ仮想化ゲストをサポートしていません。
* 外部アーキテクチャゲスト (FA) とは、ホストとは異なるハードウェアアーキテクチャ上で動作するゲストのことです。
例えば、x86ホスト上でARMゲストを実行する場合などです。Xenはこの種の仮想化をサポートしていません。なぜなら、エミュレーションやバイナリ変換が必要となるためですが、これらは非常に複雑で時間のかかる技術だからです3。

* エミュレートされたゲストとは、ホストまたは別のプラットフォームのハードウェアを模倣するソフトウェアエミュレータ上で実行されるゲストのことです。たとえば、QEMUエミュレータ上でWindowsゲストを実行する場合などです。Xenは、ハイパーバイザではなくエミュレータが仮想プラットフォームを提供するため、このタイプの仮想化をサポートしていません。Xenは、HVMゲストの一部のデバイスをエミュレートするためにQEMUを使用できますが、プラットフォーム全体をエミュレートすることはできません14。
* コンテナ仮想化ゲストとは、ホストや他のゲストと共有カーネル上で実行され、名前空間とcgroupを使用して分離されるゲストのことです。たとえば、Dockerコンテナ上でLinuxゲストを実行する場合などです。Xenはこの種の仮想化をサポートしていません。これは、ゲストオペレーティングシステムがホストカーネルと互換性がある必要があり、ハイパーバイザベースの仮想化と同じレベルの分離とセキュリティを提供しないためです56。
:
Xenプロジェクトソフトウェアの概要 - Xen
仮想化拡張機能を備えた Xen ARM - Xen
Xenプロジェクト初心者ガイド - Xen
QEMU - Xen
Dockerの概要 | Dockerドキュメント
コンテナとは？ | アプリケーションのコンテナ化 | VMware

質問**: 18**
Xen Domain-0に割り当てられるメモリ量を制限するオプションdom0_memを使用するには、このオプションはどこで指定する必要がありますか？
A. Xen が起動される際のブートローダー構成。
B. Xenのグローバル設定ファイルのいずれか。
C. ドメイン0カーネルがビルドされる際の.configファイル内。
D. Xen の起動時に、設定ファイル /etc/xen/Domain-0.cfg で設定します。
E. Xen をビルドする際の Makefile 内。
正解**: A (**コメントを発表する**)**
説明
オプションdom0_memは、Xenにおいて最初に起動し、非特権ドメイン（DomU）を管理する特権ドメインであるDomain-0の初期メモリサイズと最大メモリサイズを設定するために使用されます。Xenの起動時には、GRUBやGRUB2などのブートローダー設定でオプションdom0_memを指定する必要があります。これにより、Domain-0カーネルは、起動時のメモリ量に基づいて、メモリメタデータとネットワーク関連パラメータを格納するためのメモリを割り当てることができます。ブートローダー構成でオプションdom0_memが指定されていない場合、Domain-0はデフォルトでホストシステムの利用可能なメモリすべてを使用するため、パフォーマンスやセキュリティの問題が発生する可能性があります。参考文献：
* Xen Dom0sのCPUとメモリの管理
* Xenプロジェクトのベストプラクティス
* Dom0メモリ - メモリがまだ使われていない場所

質問: **19**
VirtualBoxに関して、以下の記述のうち正しいものはどれですか？
A. これは、最初の通常のオペレーティングシステムが起動する前に起動される、特別なカーネルとして設計されたハイパーバイザーです。
B. ゲストオペレーティングシステムとしてLinuxのみをサポートしており、仮想マシン内でWindowsを実行することはできません。
C. 仮想マシンのディスクイメージを仮想化ホストのブロックデバイス上にローカルに保存できないため、専用の共有ストレージが必要です。
D. 仮想マシンを管理するためのグラフィカルユーザーインターフェースとコマンドラインツールの両方を提供します。
E. Linux専用であり、現在実行中のLinuxカーネルのソースコードが利用可能である必要があります。
正解: **(正解を表示します)**
VirtualBoxはホスト型ハイパーバイザであり、既存のオペレーティングシステム上でアプリケーションとして動作し、最初の通常のオペレーティングシステムが起動する前に起動される特別なカーネルとしては動作しません1。VirtualBoxは、Windows、Linux、Solaris、OS/2、OpenBSD1など、多数のゲストオペレーティングシステムをサポートしています。VirtualBoxは、仮想マシンのディスクイメージを仮想化ホストのブロックデバイス、ネットワーク共有、またはiSCSIターゲットにローカルに保存できるため、専用の共有ストレージを必要としません1。VirtualBoxは、仮想マシンを管理するためのグラフィカルユーザーインターフェイス（GUI）とコマンドラインツール（VBoxManage）の両方を提供します1。VirtualBoxは、Windows、Linux、macOS、Solarisホストで利用可能であり1、現在実行中のLinuxカーネルのソースコードが利用可能である必要はありません。参考文献：
* Oracle VM VirtualBox: 機能概要

質問: **20**
以下のコマンドを使用してデータコンテナを設定した後：
docker create -v /data --name datastore debian /bin/true
データストアコンテナと/datavolumeを共有する新しいコンテナを新たに起動するにはどうすればよいですか？
A. docker run --volume-backend datastore -v /data --name service debian bash
B. docker run -v /data --name service debian bash
C. docker run --volumes-from datastore --name service debian bash
D. docker run -v datastore:/data --name service debian bash
E. docker run --share-with datastore --name service debian bash
正解: **(正解を表示します)**

質問: **21**
Xenは、以下のゲストシステムのうち、どれをサポートしていますか？（2つ選択してください。）
A. コンテナ仮想化ゲスト
B. 完全仮想化ゲスト
C. 準仮想化クエスト (PVI

D. エミュレートされたゲスト
E. 外国人建築家ゲスト FA）
正解: **(**正解を表示します**)**

質問: **22**
cloud-initは、外部ソースから取得したステータス情報と設定情報をどのディレクトリに保存しますか？
A. /var/lib/cloud/
B. /etc/cloud-init/cache/
C. /proc/sys/cloud/
D. /tmp/.cloud/
E. /opt/cloud/var/
正解: **(**正解を表示します**)**
cloud-init は /var/lib/cloud/ ディレクトリを使用して、クラウド プラットフォームのメタデータ サービスやユーザー データ ファイルなどの外部ソースから取得したステータス情報と構成情報を格納します。このディレクトリには、インスタンス、データ、ハンドラー、スクリプト、セムなど、さまざまな種類のデータ用のサブディレクトリが含まれています。インスタンス サブディレクトリには、インスタンス ID、ユーザー データ、cloud-init 構成など、現在のインスタンスに固有の情報が含まれています。データ サブディレクトリには、cloud-init が検出して使用したデータ ソースに関する情報が含まれています。ハンドラー サブディレクトリには、cloud-init が実行したハンドラーに関する情報が含まれています。
scripts サブディレクトリには、インスタンスごと、ブートごと、1 回ごと、ベンダーごとなど、ブートプロセスのさまざまな段階で cloud-init が実行するスクリプトが含まれています。sem サブディレクトリには、cloud-init がさまざまなモジュールとステージの実行ステータスを追跡するために使用するセマフォ ファイルが含まれています。参照:
* RHEL 8 の cloud-init の設定と管理 - Red Hat カスタマーポータル
* vSphere - cloud-init ユーザーが使用する Linux ファイルの場所はどこですか？

質問: **23**
次のうち、仮想化の利点はどれですか？
A. 上記のすべて
B. セキュリティの向上
C. 資源利用率の低下
D. ハードウェア管理の簡素化
正解: **D (**コメントを発表する**)**
仮想化の主要な利点の1つは、ハードウェア管理の簡素化です。仮想化は物理ハードウェアをソフトウェア定義のリソースに抽象化することで、システムの管理、割り当て、移行、保守を容易にします。
選択肢Cは、仮想化によってリソース利用率が低下するのではなく増加するため、誤りです。1つの選択肢が誤っているため、 止記のすべて」は正解になりません。セキュリティの向上も利点ですが、この文脈で最も正確な単一の回答は、ハードウェア管理の簡素化です。
したがって、正解はDです。

質問: **24**
Xenゲストドメインの仮想CD-ROMドライブ内のメディアを変更するxlコマンドのサブコマンドはどれですか？
（パスやパラメータは指定せず、サブコマンドのみを指定してください。）
解決：
block-attach - または - xl cd-insert - または - cd-eject - または - xl cd-eject - または - cd-insert - または - xl block-attach - または - block-detach - または - xl block-detach 与えられた解決策が正しいかどうかを判断してください。
A. 正解
B. 不正解
正解: **B (**コメントを発表する**)**
Xen xlツールスタックの公式ドキュメントによると、仮想CD-ROMドライブ内のメディアを変更するために使用する正しいサブコマンドはcd-insertとcd-ejectです。これらのコマンドは、Xenゲストの仮想CD-ROMデバイスにISOメディアを挿入または取り出すために特別に設計されています。
block-attachやblock-detachといったコマンドは有効なxlサブコマンドですが、これらは一般的にブロックデバイスのアタッチとデタッチに使用されるものであり、CD-ROMメディアの操作に特化したものではありません。提供されている解決策では、正しいサブコマンドを正確に指定するのではなく、誤ったコマンドや無関係なコマンドを含む複数のコマンドが列挙されています。
この解答は、正しいxlサブコマンドを正確かつ排他的に特定していないため、誤りであるとみなされます。したがって、正しい判定はBです。

質問: **25**
LXCとDockerがコンテナを作成するために使用するメカニズムは次のうちどれですか？ （3つ選択してください。）
A. Linuxの機能
B. カーネル名前空間
C. 対照群
D. POSIXACLs
E. ファイルシステムのアクセス許可
正解: **(**正解を表示します**)**
説明
LXCとDockerはどちらも、Linuxカーネルの機能を利用してアプリケーションを実行するための隔離された環境を作成するコンテナ技術です。これらが使用する主なメカニズムは次のとおりです。
* Linux 機能: これらは、プロセスが特定のシステム リソースや操作にアクセスできるように制限するためにプロセスに割り当てることができる一連の権限です。たとえば、CAP_NET_ADMIN 機能を持つプロセスは、ネットワーク インターフェイスの作成や削除などのネットワーク管理タスクを実行できます。Linux
* この機能により、コンテナは権限を制限して実行できるため、セキュリティと分離性が向上します。
カーネル名前空間 :これは、異なるプロセスに対してシステムリソースの個別のビューを作成する方法です。たとえば、マウント名前空間内のプロセスは、ホストや他の名前空間とは異なるファイルシステムレ

イアウトを持つことができます。カーネル名前空間を使用すると、コンテナはホストや他のコンテナに干渉することなく、独自のネットワークインターフェイス、プロセスID、ユーザーID、およびその他のリソースを持つことができます。
* コントロールグループ :これは、プロセスをグループ化し、リソース制限と利用状況の計測を適用する方法です。たとえば、コントロールグループは、プロセスまたはプロセスグループが使用できるCPU、メモリ、ディスクI/O、またはネットワーク帯域幅の量を制限できます。コントロールグループを使用することで、コンテナはシステムリソースを公平に分配され、ホストリソースを使い果たしてしまうことを防ぐことができます。
POSIX ACLやファイルシステムのパーミッションは、LXCやDockerがコンテナを作成する際に使用するメカニズムではありません。
これらはファイルシステム上のファイルやディレクトリへのアクセスを制御する方法であり、コンテナだけでなく、あらゆるプロセスに適用できる。
参考文献：
* LXCとDocker :どちらのコンテナプラットフォームがあなたに最適ですか？
* LXC vs Docker :2023年においてDockerが優れている理由 | UpGuard
* LXC、LXD、Dockerコンテナの違いは何ですか？
* lxc - Dockerが使用しているコンテナ実装はどれか - Unix & Linux Stack Exchange

質問: **26**
cloud-initの目的は何ですか？
A. systemdやSysV initなどの一般的なLinux inicシステムを置き換えます。
B. クラウド内の特定のコンピューティングノードにIaaSインスタンスを割り当てます。
C. クラウドにおけるロードバランサーや仮想ファイアウォールなどのインフラストラクチャサービスの構成を標準化する。
D. 複数の関連するIaaSインスタンスの作成と起動をオーケストレーションします。
E. 特定のインスタンスの構成に合わせて、IaaSインスタンスの汎用イメージを準備します。
正解: **(正解を表示します)**
説明
Cloud-initは、クラウド上のLinux仮想マシンの初回起動時に、設定を処理し、5つの段階を経て実行されるツールです。ユーザーは、インスタンスにユーザーデータを適用することで、Linux仮想マシンの初回起動時にカスタマイズを行うことができます。ユーザーデータには、スクリプト、コマンド、パッケージ、ファイル、ユーザー、グループ、SSHキーなどが含まれます。
Cloud-init は、Azure、AWS、OpenStack など、さまざまなクラウドプラットフォームやサービスと連携することもできます。cloud-init の目的は、ホスト名、ネットワーク、セキュリティ、アプリケーション設定など、特定のインスタンスの構成に合わせて、IaaS インスタンスの汎用イメージを準備することです。参考文献:
* Cloud-init - クラウドインスタンスをカスタマイズするための標準規格
* cloud-init の理解 - Azure 仮想マシン
* チュートリアル - Azure で cloud-init を使用して Linux VM をカスタマイズする - Azure 仮想マシン

質問: **27**
cloud-initは、外部ソースから取得したステータス情報と設定情報をどのディレクトリに保存しますか？
A. /var/lib/cloud/
B. /etc/cloud-init/cache/
C. /proc/sys/cloud/
D. /tmp/.cloud/
E. /opt/cloud/var/
正解: **(**正解を表示します**)**
説明
cloud-init は /var/lib/cloud/ ディレクトリを使用して、クラウド プラットフォームのメタデータ サービスやユーザー データ ファイルなどの外部ソースから取得したステータス情報と構成情報を格納します。このディレクトリには、インスタンス、データ、ハンドラー、スクリプト、セムなど、さまざまな種類のデータ用のサブディレクトリが含まれています。インスタンス サブディレクトリには、インスタンス ID、ユーザー データ、cloud-init 構成など、現在のインスタンスに固有の情報が含まれています。データ サブディレクトリには、cloud-init が検出して使用したデータ ソースに関する情報が含まれています。ハンドラー サブディレクトリには、cloud-init が実行したハンドラーに関する情報が含まれています。
scripts サブディレクトリには、インスタンスごと、ブートごと、1 回ごと、ベンダーごとなど、ブートプロセスのさまざまな段階で cloud-init が実行するスクリプトが含まれています。sem サブディレクトリには、cloud-init がさまざまなモジュールとステージの実行ステータスを追跡するために使用するセマフォ ファイルが含まれています。参照:
* RHEL 8 の cloud-init の設定と管理 - Red Hat カスタマーポータル
* vSphere - cloud-init ユーザーが使用する Linux ファイルの場所はどこですか？

質問: **28**
KVMドメインにおいて、libvirtによって制限できるリソースは次のうちどれですか？ （2つ選択してください。）
A. CPUライムの量
B. 利用可能なメモリのサイズ
C. ドメイン内で許可されているファイルシステム
D. 実行中のプロセス数
E. 利用可能なファイル数
正解: **A,B (**コメントを発表する**)**
説明
Libvirtは、KVM、Xen、LXCなどのさまざまな仮想化技術を管理するための共通APIを提供するツールキットです。Libvirtを使用すると、仮想マシン（ドメインとも呼ばれます）のCPU、メモリ、ディスク、ネットワーク、その他のリソースなど、さまざまな側面を構成および制御できます。LibvirtでKVMドメインに対して制限できるリソースには、次のようなものがあります。
* CPU時間：Libvirtでは、ドメインが使用できる仮想CPU（vCPU）の数、CPUモード、モデル、トポロジ、チューニングパラメータを指定できます。また、CPUシェア、クォータ、期間を設定して、ドメインが消費できるCPU時間の相対的または絶対的な量を制御することもできます。さらに、vCPUを物理CPUまたは

NUMAノードに固定して、パフォーマンスと分離性を向上させることもできます。これらの設定は、ドメインXMLファイルの<cpu>要素と<cputune>要素12で構成できます。
* 使用可能なメモリのサイズ: Libvirt では、ドメインが使用できるメモリの量、メモリのバックアップ、チューニング、および NUMA ノードのパラメータを指定できます。また、メモリのハード リミットとソフト リミット、スワップのハード リミット、および最小保証を設定して、ドメインのメモリ割り当てと再利用ポリシーを制御することもできます。これらの設定は、ドメイン XML ファイルの <memory>、<memoryBacking>、および <memtune> 要素で構成できます 13。
質問に記載されているその他のリソースは、KVM ドメインの場合、libvirt によって直接制限されるものではありません。ドメインで許可されるファイルシステムは、ドメインに接続されているディスクおよびファイルシステム デバイスによって決定され、これらはドメイン XML ファイルの <disk> および <filesystem> 要素で構成できます 14。実行中のプロセス数と使用可能なファイル数は、ドメインのオペレーティングシステムとファイルシステムによって決定され、libvirt によって制御されるものではありません。
参考文献：
* libvirt: ドメインXML形式
* CPU割り当て
* メモリ割り当て
ハードディスク、フロッピーディスク、CD-ROM

質問: **29**
以下のコンポーネントのうち、Dockerアーキテクチャにおいて不可欠なものはどれですか？（該当するものをすべて選択してください）
A. ドッカーデーモン
B. Dockerレジストリ
C. Dockerクライアント
D. Docker Compose
正解: **(**正解を表示します**)**

質問: **30**
ハードウェア仮想化拡張機能を備えたQEMUを使用するためにロードする必要があるLinuxカーネルモジュールに関して、正しい記述はどれですか？
A. 仮想マシンのコンソールがホストシステムの物理コンソールに接続される場合にのみ、ホストシステムのカーネルにロードする必要があります。
B. QEMUベアメタルハイパーバイザと連携し、追加の仮想マシンの起動をトリガーする必要があるため、最初の仮想マシンのカーネルにロードする必要があります。
C. ホストシステムのCPUの可視化拡張機能を使用するには、ホストシステムのカーネルにロードする必要があります。
D. ホストシステムのファイルシステムからファイルやディレクトリにアクセスする各仮想マシンのカーネルにロードする必要があります。

E. QEMU に必要な Para 仮想化を提供するために、各仮想マシンのカーネルにロードする必要があります。
正解: **(**正解を表示します**)**

質問: **31**
virshコマンドの中で、現在ホスト上で実行されている仮想マシンの一覧を表示するコマンドはどれですか？
A. 私は
B. list-vm
C. リスト
D. ショー
E. すべて表示
正解: **(**正解を表示します**)**
コマンド virsh list は、現在のホスト上で実行中のすべてのドメイン (VM) を一覧表示するために使用されます。
--all を使用すると、アクティブなドメインと非アクティブなドメインの両方を一覧表示できます。その他のオプションは、有効な virsh コマンドではありません。コマンド virsh list は、実行中のすべてのドメイン (VM) を一覧表示する基本的なコマンドです。--all オプションを追加することで、構成済みのすべての VM を一覧表示することもできます。これは、後続のコマンドで使用できる、ターゲット ハイパーバイザで構成されているすべての VM を表示する場合に便利です 1。参照:
* 1: コマンドラインで VM を管理するための 8 つの Linux virsh サブコマンド | システム管理者を有効にする。

有効的な**305-300**問題集はJPNTest.com提供され、**305-300**試験に合格することに役に立ちます！JPNTest.comは今最新**305-300**試験問題集を提供します。JPNTest.com 305-300試験問題集はもう更新されました。ここで**305-300**問題集のテストエンジンを手に入れます。最新版のアクセス、https://www.jpntest.com/shiken/305-300-mondaishu　**125**問、３０%ディスカウント、特別な割引コード: **JPNshiken**」

質問: **32**
libvirtドメイン定義内の<os>要素のfirmware属性で有効な値は次のうちどれですか？ 2つ選択してください。）
A. SCSI
B. virtio
C. efi
D. バイオ
E. PCIe
正解: **C,D (**コメントを発表する**)**

libvirtドメイン定義の<os>要素のファームウェア属性は、仮想マシンの起動に使用されるファームウェアの種類を指定します。この属性の有効な値はefiとbiosで、それぞれ拡張ファームウェアインターフェイス（EFI）と基本入出力システム（BIOS）に対応します。EFIはBIOSよりも多くの機能とセキュリティをサポートする新しい標準規格で、セキュアブートや高速な起動時間などが含まれます。BIOSは古い標準規格ですが、ほとんどのハイパーバイザーとオペレーティングシステムで広く互換性がありサポートされています。その他の値であるscsi、virtio、pcieはファームウェアとは関係なく、仮想マシンで使用できるさまざまな種類のデバイスまたはドライバに関連しています。参考文献 :1（ファームウェア列挙型を検索）

質問: **33**
コマンドlxc-checkpointに関する以下の記述のうち、正しいものはどれですか？
A. コンテナのクローンを作成します。
B. コンテナのメモリ消費量が2倍になります。
C. 停止中のコンテナでのみ機能します。
D. コンテナの状態をファイルに書き込みます。
E. 既存のコンテナに基づいてコンテナイメージを作成します。
正解: **(正解を表示します)**
lxc-checkpoint コマンドは、コンテナのチェックポイントと復元に使用されます。コンテナのチェックポイントとは、メモリ、プロセス、ファイルディスクリプタ、ネットワーク接続など、コンテナの状態をファイルまたはディレクトリに保存することです。コンテナの復元とは、保存された状態からコンテナを再開し、停止していなかった状態に戻すことです。コンテナのチェックポイントと復元は、ライブマイグレーション、バックアップ、デバッグ、スナップショットなど、さまざまな用途に役立ちます。lxc-checkpoint コマンドの構文は次のとおりです。
lxc-checkpoint {-n 名前} {-D パス} [-r] [-s] [-v] [-d] [-F]
選択肢は以下のとおりです。
* -n name: チェックポイントを作成または復元するコンテナの名前を指定します。
* -D path: チェックポイントデータがダンプまたは復元されるファイルまたはディレクトリへのパスを指定します。
* -r、--restore: コンテナのチェックポイントをダンプする代わりに復元します。このオプションは -s と互換性がありません。
* -s、--stop: ダンプ後にコンテナを停止します（オプション）。このオプションは -r と互換性がありません。
* -v、--verbose: criu の詳細なログ記録を有効にします。-r を指定した場合のみ利用可能です。
* -d、--daemon: コンテナをバックグラウンドで復元します (これがデフォルトです)。-r を指定した場合のみ利用可能です。
* -F、--foreground: コンテナをフォアグラウンドで復元します。-r を指定した場合のみ利用可能です。
コマンド lxc-checkpoint は、CRIU (Checkpoint/Restore In Userspace) ツールを使用してチェックポイントと復元操作を実行します。CRIU は、実行中のアプリケーション (またはその一部) を一時停止し、ファイルの集合としてハードドライブにチェックポイントを作成できるソフトウェアです。その後、これらのファイルを使用して、一時停止した時点からアプリケーションを復元して実行できます。

lxc-checkpoint コマンドに関するその他の記述は正しくありません。このコマンドはコンテナのクローンやイメージを作成するものではなく、コンテナのメモリ使用量を倍増させるものでもありません。指定されたオプションによっては、実行中のコンテナと停止中のコンテナの両方で動作します。参考資料：
* Linuxコンテナ - LXC - マニュアルページ - lxc-checkpoint.12
* lxc-checkpoint(1) - Linux マニュアルページ - man7.org3
* CRIU4

質問: **34**
cloud-initprocessがユーザーデータから直接処理できるデータの種類は次のうちどれですか？ 3つ選択してください。）
A. YAMLでのcloud-config宣言
B. インポートするURLのリスト
C. 起動に使用するISOイメージ
D. 実行するシェルスクリプト
E. 実行するBase64エンコードされたバイナリファイル
正解: **(正解を表示します)**

質問: **35**
次のうち、コンテナオーケストレーションプラットフォームはどれですか？
A. Docker Compose
B. ドッカー ·スウォーム
C. LXC
D. Kubernetes
正解: **B,D (コメントを発表する)**
コンテナオーケストレーションプラットフォームは、複数のホストにわたるコンテナ化されたアプリケーションのデプロイ、スケーリング、ネットワーク構築、および管理を担当します。コンテナ化に関するドキュメントによると、Docker SwarmとKubernetesは、代表的なコンテナオーケストレーションプラットフォームとして認識されています。
Docker Swarmは、クラスタリング、サービスディスカバリ、ロードバランシング、ローリングアップデートを提供する、Dockerのネイティブなオーケストレーションソリューションです。Kubernetesは、高度なスケジューリング、自己修復、水平スケーリング、宣言型構成を提供する、業界標準のオーケストレーションプラットフォームです。
Docker Composeは完全なオーケストレーションプラットフォームとは見なされていません。主に開発やテストを目的として、単一ホスト上で複数のコンテナアプリケーションを定義して実行するために設計されています。LXCはコンテナランタイムおよびシステムコンテナ技術であり、オーケストレーションフレームワークではありません。
コンテナのドキュメントでは、オーケストレーションプラットフォームとランタイムおよびローカル開発ツールが明確に区別されています。したがって、正解はBとDです。

質問: **36**

virshコマンドの中で、現在ホスト上で実行されている仮想マシンの一覧を表示するコマンドはどれですか？
A. リスト
B. list-vm
C. 私は見る
D. ショー
E. すべて表示
正解: **A (**コメントを発表する**)**

質問: **37**
KVM仮想マシン内でLinuxを起動するために直接Linuxブートを使用する場合、Linuxカーネルのパラメータを指定するために使用するKVMパラメータはどれですか？
（値やパラメータは指定せず、オプション名のみを指定してください。）
正解:
-追加
Explanation:
KVM/QEMUでカーネルを直接起動する場合、-appendoptionを使用してカーネルコマンドラインパラメータをLinuxカーネルに渡します。これにより、管理者はルートファイルシステムの場所、コンソール設定、デバッグフラグなどの起動時オプションを指定できます。
QEMUの公式ドキュメントには、カーネル引数の正しいパラメータとして「append」が記載されているため、この回答は正しいです。

質問: **38**
クラウドインスタンスの設定に使用できるcloud-initデータソースの種類はどれですか？（該当するものをすべて選択してください）
A. クラウドベースの構成管理ツール
B. メタデータサービス
C. QRコード
D. ユーザーデータスクリプト
正解: **B,D (**コメントを発表する**)**
Cloud-initは、クラウドインスタンス向けの広く利用されている初期化システムであり、初回起動時に設定を行うことができます。
cloud-initのドキュメントによると、インスタンスの設定に使用される主なデータソースは、メタデータサービスとユーザーデータスクリプトの2つです。
メタデータサービスは、ホスト名、インスタンスID、ネットワークの詳細、SSHキーなどのインスタンス固有の情報を提供します。ユーザーデータスクリプトを使用すると、管理者はカスタム構成手順を提供できます。これは通常、シェルスクリプトまたはクラウド構成YAMLとして記述されます。
AnsibleやPuppetのようなクラウドベースの構成管理ツールは、cloud-initのデータソースではありません。代わりに、cloud-initはこれらのツールをインストールしたり、起動したりすることができます。QRコードはcloud-initでサポートされているデータソースではありません。

仮想化とクラウドに関するドキュメントには、メタデータとユーザーデータがサポートされている構成メカニズムとして明確に記載されています。したがって、正解はBとDです。

質問: **39**
IaaSとは何ですか？
A. 情報サービス
B. サービスとしてのインテリジェンス
C. サービスとしての統合
D. サービスとしてのインスタンス
E. サービスとしてのインフラストラクチャ
正解: **(**正解を表示します**)**
IaaSは、必要なコンピューティング、ストレージ、ネットワークリソースをオンデマンドで従量課金制で提供するクラウドコンピューティングサービスの一種です。IaaSは、SaaS（Software as a Service）、PaaS（Platform as a Service）、サーバーレス12とともに、4種類のクラウドサービスの1つです。IaaSを利用することで、企業はインフラストラクチャを自社で調達、構成、管理する必要がなくなり、使用した分だけ料金を支払うことができます23。IaaSプロバイダーの例としては、Microsoft Azure、Google Cloud、Amazon Web Servicesなどがあります。

質問: **40**
cloud-initの目的は何ですか？
A. 複数の関連するIaaSインスタンスの作成と起動をオーケストレーションします。
B. systemdやSysV initなどの一般的なLinux inicシステムを置き換えます。
C. クラウドにおけるロードバランサーや仮想ファイアウォールなどのインフラストラクチャサービスの構成を標準化する。
D. クラウド内の特定のコンピューティングノードにIaaSインスタンスを割り当てます。
E. 特定のインスタンスの構成に合わせて、IaaSインスタンスの汎用イメージを準備します。
正解: **(**正解を表示します**)**

質問: **41**
QEMU仮想マシンのCPUに関して、次のうち正しいものはどれですか？（2つ選択してください。）
A. QEMU仮想マシンのCPUアーキテクチャは、ホストシステムのアーキテクチャとは独立しています。
B. 各QEMU仮想マシンは、1つのコアを持つCPUを1つしか持つことができません。
C. 各QEMU仮想マシンには、専用の物理CPUコアを1つ予約する必要があります。
D. QEMUは仮想CPUの概念を使用して、仮想マシンを物理CPUにマッピングします。
E. QEMU仮想マシンは、SMPシステムを実行するために複数の仮想CPUをサポートしています。
正解: **(**正解を表示します**)**
説明
QEMU 仮想マシンの CPU アーキテクチャは、ホスト システムのアーキテクチャとは独立しています。QEMU は、ホスト システムのアーキテクチャに関係なく、x86、ARM、Alpha、SPARC など、多くの CPU アーキテクチャをエミュレートできます 1。これにより、QEMU はホスト システムのハードウェア

と互換性のないゲスト オペレーティング システムを実行できます。したがって、オプション A が正解です。QEMU 仮想マシンは、SMP システムを実行するために複数の仮想 CPU をサポートしています。QEMU は、仮想 CPU (vCPU) の概念を使用して、仮想マシンを物理 CPU にマッピングします。各 vCPU は、物理 CPU コアで実行されるスレッドです。QEMU では、ユーザーが各仮想マシンの vCPU の数と CPU モデルを指定できます。QEMU は、複数の vCPU を持つ SMP システムと、1 つの vCPU を持つシングル プロセッサ システムを実行できます 2。したがって、オプション E も正解です。他のオプションは、QEMU 仮想マシンの CPU を説明していないため、不正解です。オプション B は、QEMU 仮想マシンが複数のコアを持つ複数の CPU を持つことができるため、不正解です。選択肢Cは誤りです。QEMUは仮想マシンごとに専用の物理CPUコアを必要としません。QEMUは負荷とスケジューリングポリシーに応じて、複数の仮想マシン間で物理CPUコアを共有できます。
選択肢Dは誤りです。QEMUでは仮想マシンのプロセッサを指すのにCPUではなくvCPUという用語を使用します。参考文献：
* QEMUとVirtualBox：違いは何？ - LinuxConfig.org
* QEMU / KVM CPU モデル構成 - QEMU ドキュメント
* 概要 - QEMU ドキュメント
* Qemu/KVM仮想マシン - Proxmox仮想環境

質問: **42**
.dockerignoreファイルの目的は何ですか？
A. 派生イメージを構築する際に除外すべき、Dockerイメージ内に存在するファイルを一覧表示します。
B. Dockerイメージをビルドする際に、DockerがDockerデーモンに送信しないファイルを指定します。
C. Dockerによって提供されるボリュームとポートを無視する必要があるコンテナのルートファイルシステムに存在します。
D. Dockerがコンテナに自動的にアタッチしてはならないボリュームの最上位ディレクトリに配置する必要があります。
E. Dockerイメージをビルドする際に、Dockerfileのどの部分を無視するかを指定します。
正解: **(**正解を表示します**)**
.dockerignore ファイルの目的は、Docker イメージをビルドする際に Docker が Docker デーモンに送信しないファイルを指定することです。.dockerignore ファイルは、ビルド コンテキスト（Dockerfile で使用できるファイルとフォルダーのセット）から除外するファイルまたはディレクトリのリストを含むテキスト ファイルです。.dockerignore ファイルを使用することで、サイズが大きいファイルや機密情報を含むファイル、または Docker イメージに関係のないファイルやディレクトリをデーモンに送信することを回避でき、ビルド プロセスの効率とセキュリティを向上させることができます。他のオプションは、.dockerignore ファイルの機能を説明していないため、誤りです。オプション A は、.dockerignore ファイルが Docker イメージに存在するファイルには影響せず、ビルド中にデーモンに送信されるファイルのみに影響するため、誤りです。オプション C は、.dockerignore ファイルの機能を説明していないため、誤りです。
dockerignore ファイルはコンテナのルートファイルシステムには存在せず、Dockerfile と同じディレクトリに存在します。オプション D は、.dockerignore ファイルが Docker がコンテナにアタッチするボ

リュームには影響せず、ビルドコンテキストに含まれるファイルのみに影響するため誤りです。オプション E は、.dockerignore ファイルが実行される Dockerfile の部分には影響せず、Dockerfile で使用可能なファイルのみに影響するため誤りです。
参考文献：
* .dockerignoreファイルとは何ですか？また、なぜ使用すべきなのでしょうか？
* Dockerfile リファレンス | Docker ドキュメント
* .dockerignore の使い方とその重要性 - Shisho Cloud

質問: **43**
acgroupdirectory内のどのファイルに、このcgroupに属するプロセスの一覧が含まれていますか？
A. 樽
B. 被験者
C. メンバー
D. pids
E. プロシージャ
正解: (正解を表示します)

質問: **44**
ハードウェア仮想化拡張機能を備えたQEMUを使用するためにロードする必要があるLinuxカーネルモジュールに関して、正しい記述はどれですか？
A. 仮想マシンのコンソールをホストシステムの物理コンソールに接続する場合にのみ、ホストシステムのカーネルにロードする必要があります。
B. ホストシステムのファイルシステムからファイルやディレクトリにアクセスする各仮想マシンのカーネルにロードする必要があります。
C. ホストシステムのCPUの可視化拡張機能を使用するには、ホストシステムのカーネルにロードする必要があります。
D. QEMUベアメタルハイパーバイザと連携し、追加の仮想マシンの起動をトリガーする必要があるため、最初の仮想マシンのカーネルにロードする必要があります。
E. QEMU に必要な Para 仮想化を提供するために、各仮想マシンのカーネルにロードする必要があります。
正解: (正解を表示します)
QEMUでハードウェア仮想化拡張機能を使用するためにロードする必要があるLinuxカーネルモジュールは、KVM（Kernel-based Virtual Machine）です。KVMは、ユーザー空間プログラム（QEMUなど）がさまざまなプロセッサ（Intel VTやAMD-Vなど）のハードウェア仮想化機能を利用できるようにする完全な仮想化ソリューションです。KVMは、コア仮想化インフラストラクチャを提供するロード可能なカーネルモジュールkvm.koと、プロセッサ固有のモジュールkvm-intel.koまたはkvm-amd.koで構成されています。ホストシステムのCPUの仮想化拡張機能を使用するには、KVMをホストシステムのカーネルにロードする必要があります。これにより、QEMUは、それぞれが独自の仮想化ハードウェアを持つ、変更されていないLinuxまたはWindowsイメージを使用して複数の仮想マシンを実行できます。KVMはQEMUに統合されているため、各仮想マシンまたは最初の仮想マシンのカーネルにロードする必要はありません。KVMは

準仮想化も必要としません。準仮想化とは、ゲストオペレーティングシステムを変更してハイパーバイザと直接通信させ、エミュレーション層をバイパスする技術です。参考文献：
* 機能/KVM - QEMU
* カーネルベースの仮想マシン
* Red Hat Enterprise Linux 8 (2023) における KVM 仮想化

質問: **45**
コマンドvagrantinitの目的は何ですか？
A. Vagrantボックスをダウンロードします。
B. LinuxホストにVagrantをインストールします。
C. 実行中のボックス内でプロビジョニングツールを実行します。
D. Vagrantの設定ファイルを作成します。
E. Vagrantボックスを起動します。
正解: **(**正解を表示します**)**

質問: **46**
コンテナ仮想化の文脈において、機能（capabilities）の目的は何ですか？
A. 潜在的に危険なシステムコールを、コンテナ仮想化によって提供されるエミュレーションレイヤーにマッピングします。
B. コンテナが消費できるディスク容量を制限します。
C. 複数のコンテナに存在するファイルをキャッシュするために、メモリ重複排除を有効にします。
D. 一般ユーザーが管理者権限でコンテナを起動できるようにする。
E. コンテナに違反する可能性のあるアクションをプロセスが実行することを防止します。
正解: **(**正解を表示します**)**
説明
ケーパビリティは、Linux できめ細かなアクセス制御を実現する方法です。ケーパビリティは、プロセスが持つことができる権限を定義するフラグのセットです。デフォルトでは、プロセスは親のケーパビリティを継承しますが、一部のケーパビリティはプロセス自身またはカーネルによって削除または追加できます。コンテナ仮想化のコンテキストでは、ケーパビリティは、ホストのデバイスへのアクセス、ファイルシステムのマウント、システム時刻の変更、他のプロセスの強制終了など、コンテナを侵害する可能性のあるアクションをプロセスが実行できないようにするために使用されます。ケーパビリティを使用すると、コンテナは権限を制限して実行できるため、コンテナ環境のセキュリティと分離が強化されます。たとえば、Docker は、コンテナ内で実行されるプロセスに付与されるデフォルトのケーパビリティセットを使用し、必要に応じてユーザーがケーパビリティを追加または削除できるようにします 12。
参考文献：
* 機能 | Docker ドキュメント1
* Linuxの機能 :コンテナでそれらを活用する方法2

有効的な**305-300**問題集はJPNTest.com提供され、**305-300**試験に合格することに役に立ちます！JPNTest.comは今最新**305-300**試験問題集を提供します。JPNTest.com 305-300試験問題集はもう更新されました。ここで**305-300**問題集のテストエンジンを手に入れます。最新版のアクセス、https://www.jpntest.com/shiken/305-300-mondaishu　**125**問、３０%ディスカウント、特別な割引コード: **JPNshiken**」

質問: **47**
コマンドvirsh vol-list vmsを実行すると、次のエラーが返されます。
エラー: プール 'vms'の取得に失敗しました
エラー: ストレージ プールが見つかりません: 一致する名前 'vms' のストレージ プールがありません。
ディレクトリ /vms が存在する場合、以下のどのコマンドでこの問題を解決できますか?
A. dd if=/dev/zero of=/vms bs=1 count=0 flags=name:vms
B. libvirt-poolctl new --name=/vms --type=dir --path=/vms
C. qemu-img pool vms:/vms
D. virsh pool-create-as vms dir --target /vms
E. touch /vms/.libvirtpool
正解: **(**正解を表示します**)**
説明
コマンド `virsh pool-create-as vms dir --target /vms` は、ターゲットディレクトリ /vms12 を持つ、タイプ dir の vms という名前の一時ストレージプールを作成して起動します。このコマンドは、既存のディレクトリ /vms をストレージプールとして libvirt から認識できるようにするため、「ストレージプールが見つかりません」というエラーを解決します。他のコマンドは、以下の理由で無効です。
* dd if=/dev/zero of=/vms bs=1 count=0 flags=name:vms は有効なコマンド構文ではありません。dd コマンドは flags 引数を受け付けず、出力ファイル /vms はディレクトリではなく通常のファイルである必要があります 3。
* libvirt-poolctl new --name=/vms --type=dir --path=/vms は有効なコマンド名ではありません。libvirt パッケージ 4 には libvirt-poolctl というコマンドは存在しません。
* qemu-img pool vms:/vms は有効なコマンド構文ではありません。qemu-img コマンドには pool サブコマンドがなく、vms:/vms 引数は有効なイメージ指定ではありません 5。
* touch /vms/.libvirtpool はストレージ プールを作成するための有効なコマンドではありません。touch コマンドは空のファイルを作成するだけで、.libvirtpool ファイルは libvirt によってストレージ プール構成ファイルとして認識されません 6。
参考文献：
* 1: virsh - pool-define-as と pool-create-as の違い - Stack Overflow
* 2: dd(1) - Linux マニュアルページ - man7.org
* 3: 12.3.3. virsh を使用したディレクトリベースのストレージ プールの作成 - Red Hat カスタマー ポータル
* 4: libvirt - Linux マニュアルページ (3)
* 5: qemu-img(1) - Linux マニュアルページ - man7.org

* 6: touch(1) - Linux マニュアルページ - man7.org

質問: **48**
空欄を埋める
実行中のLXCコンテナ内で新しいプロセスを開始するLXCコマンドは何ですか？（パスやパラメータは指定せず、コマンドのみを指定してください。）
正解:
lxc-attach
説明
lxc-attach コマンドを使用すると、実行中の LXC コンテナ 12 内で新しいプロセスを開始できます。このコマンドは、コンテナ名を引数として受け取り、オプションでコンテナ内で実行するコマンドを指定します。コマンドが指定されていない場合は、コンテナ 1 内に新しいシェルが作成されます。たとえば、myContainer という名前のコンテナのホーム ディレクトリ内のすべてのファイルを一覧表示するには、次のようにします。
lxc-attach -n myContainer - ls -lh /home
参考文献：
* 1: 実行中の LXC 内でコマンドを実行する - Unix & Linux Stack Exchange

質問: **49**
QEMUはユーザーネットワークにおいて、以下のサービスのうちどれを提供できますか？（3つ選択してください。）
A. AppleTalk
B. CIFS
C. DHCP
D. BGP
E. TFTP
正解: **(正解を表示します)**

質問: **50**
コマンドvirsh vol-list vmsを実行すると、次のエラーが返されます。
エラー: プール「vms」の取得に失敗しました
エラー: ストレージ プールが見つかりません: 一致する名前 'vms' のストレージ プールがありません。
ディレクトリ /vms が存在する場合、以下のどのコマンドでこの問題を解決できますか?
A. dd if=/dev/zero of=/vms bs=1 count=0 flags=name:vms
B. libvirt-poolctl new --name=/vms --type=dir --path=/vms
C. qemu-img pool vms:/vms
D. virsh pool-create-as vms dir --target /vms
E. touch /vms/.libvirtpool
正解: **D (コメントを発表する)**

コマンド `virsh pool-create-as vms dir --target /vms` は、ターゲットディレクトリ /vms12 を持つ、タイプ dir の vms という名前の一時ストレージプールを作成して起動します。このコマンドは、既存のディレクトリ /vms をストレージプールとして libvirt から認識できるようにするため、「ストレージプールが見つかりません」というエラーを解決します。他のコマンドは、以下の理由で無効です。
* dd if=/dev/zero of=/vms bs=1 count=0 flags=name:vms は有効なコマンド構文ではありません。dd コマンドは flags 引数を受け付けず、出力ファイル /vms はディレクトリではなく通常のファイルである必要があります 3。
* libvirt-poolctl new --name=/vms --type=dir --path=/vms は有効なコマンド名ではありません。libvirt パッケージ 4 には libvirt-poolctl というコマンドは存在しません。
* qemu-img pool vms:/vms は有効なコマンド構文ではありません。qemu-img コマンドには pool サブコマンドがなく、vms:/vms 引数は有効なイメージ指定ではありません 5。
* touch /vms/.libvirtpool はストレージ プールを作成するための有効なコマンドではありません。touch コマンドは空のファイルを作成するだけで、.libvirtpool ファイルは libvirt によってストレージ プール構成ファイルとして認識されません 6。
参考文献：
1: virsh - pool-define-as と pool-create-as の違い - Stack Overflow
2: dd(1) - Linux マニュアルページ - man7.org
3: 12.3.3. virsh を使用したディレクトリベースのストレージ プールの作成 - Red Hat カスタマー ポータル
4: libvirt - Linux マニュアルページ (3)
5: qemu-img(1) - Linux マニュアルページ - man7.org
6: touch(1) - Linux マニュアルページ - man7.org

質問**: 51**
IaaSクラウドにおいて、オペレーティングシステムとソフトウェアを搭載した新しいコンピューティングインスタンスをプロビジョニングする一般的な方法は何ですか？
A. 各新規インスタンスはLinuxディストリビューションのインストールメディアに接続され、SSH経由でログインすることでインストーラーにアクセスできます。
B. 各新規インスタンスは、オペレーティングシステム、ソフトウェア、および特定の目的のためのデフォルト設定を含むイメージファイルに基づいて作成されます。
C. 新しいインスタンスはそれぞれ、現在実行中の別のインスタンスのクローンであり、元のインスタンスのすべてのソフトウェア、データ、および状態を含みます。
D. 新しいインスタンスはそれぞれ、プロビジョニングを開始したコンピュータとVPN経由で接続され、そのマシンからPXEブートを試みます。
E. 各新規インスタンスには、仮想CDから実行される最小限のライブシステムが含まれており、管理者はこれを基にターゲットオペレーティングシステムを展開します。
正解**: (**正解を表示します**)**
IaaSクラウドでは、新しいコンピューティングインスタンスをプロビジョニングする最も一般的な方法は、オペレーティングシステムとソフトウェアがプリインストールされたイメージファイルを使用する

ことです。このイメージファイルは、マシンイメージ、仮想アプライアンス、またはテンプレートとも呼ばれます。イメージファイルは、Webサーバー、データベースサーバー、開発環境など、特定の目的に合わせてカスタマイズできます。イメージファイルは、クラウドプロバイダーまたはユーザーがアクセスできるリポジトリまたはライブラリに保存できます。新しいインスタンスが要求されると、クラウドプロバイダーはイメージファイルを仮想ディスクにコピーし、インスタンスにアタッチします。インスタンスは仮想ディスクから起動し、イメージファイルからオペレーティングシステムとソフトウェアを実行します。この方法は、新しいインスタンスごとにオペレーティングシステムとソフトウェアを最初からインストールするよりも高速で効率的です。また、同じイメージファイルを使用する複数のインスタンス間で一貫性と信頼性を確保します。参考文献：
* LPI仮想化およびコンテナ化試験の目標、トピック305.1：仮想化の概念と理論、目標：マシンイメージとテンプレートの概念を説明する
* LPI仮想化およびコンテナ化学習ガイド、第1章：仮想化の概念と理論、セクション：マシンイメージとテンプレート
* LPI LPIC-3 305 認定試験のサンプル問題と練習問題、質問 10: IaaS クラウドでは、オペレーティングシステムとソフトウェアを使用して新しいコンピューティング インスタンスをプロビジョニングする一般的な方法は何ですか?

質問: **52**
空欄を埋める
virshopコマンドのどのサブコマンドを使用すると、仮想ネットワークのXML設定をエディタで展開して、その設定を変更できますか？（パラメータは指定せず、サブコマンドのみを指定してください。）
正解:
ネット編集
説明
virsh のサブコマンドで、仮想ネットワークの XML 設定をエディタで開き、その設定を変更できるのは net-edit です。このサブコマンドは、ネットワーク名または UUID をパラメータとして受け取り、$EDITOR シェル変数で指定されたデフォルトのエディタでネットワーク XML ファイルを開きます。ネットワーク設定への変更は、エディタを保存して終了するとすぐに適用されます。
参考文献：
* 1: net-edit - libvirt。

質問: **53**
Vagrantはどのようなシナリオで開発者にとって最も有益となるでしょうか？
A. ネットワークインフラストラクチャを管理する際
B. アプリケーションをクラウドにデプロイする場合
C. 大規模な生産環境で作業する場合
D. アプリケーションをローカルで開発およびテストする場合
正解: **D (**コメントを発表する**)**
Vagrantは、再現可能な開発環境の作成と管理を簡素化するために設計されたツールです。仮想化およびコンテナ化に関するドキュメントによると、Vagrantはアプリケーションをローカルで開発およびテスト

する際に最も効果を発揮します。開発者は宣言型の構成ファイルを使用して仮想マシンの構成を定義できるため、開発チーム間で一貫した環境を確保できます。
VagrantはVirtualBox、KVM、VMwareなどの仮想化プロバイダーと連携し、開発者が本番環境を忠実に再現した軽量で使い捨て可能な仮想マシンを起動できるようにします。これにより、開発環境の標準化が図られ、「自分のマシンでは動くのに、Vagrantでは動かない」といったよくある問題を軽減できます。
Vagrantは、ネットワークインフラストラクチャ、大規模な本番環境、または直接的なクラウドデプロイメントの管理を目的としたものではありません。クラウドプロバイダーとの連携は可能ですが、主な用途は本番運用ではなく、ローカルでの開発およびテストです。
仮想化に関する注記では、Vagrantはインフラ自動化ツールを置き換えるのではなく、補完する開発者向けツールであることが強調されています。したがって、オプションDはVagrantの最も適切で文書化された使用例を正確に反映しています。

質問: **54**
VagrantとDockerの両方で提供される機能はどれですか？（3つ選択してください。）
A. どちらもホストファイルシステムからゲストにディレクトリを共有できます。
B. デフォルトでは、どちらもシステムイメージを仮想マシンではなくコンテナとして起動します。
C. どちらも必要なベースイメージをダウンロードできます。
D. どちらもベース画像に変更を加えることができます。
E. デフォルトでは、両方ともコンテナではなく仮想マシンとしてシステムイメージを起動します。
正解: **(**正解を表示します**)**
VagrantとDockerはどちらも、ホストのファイルシステムからゲストにディレクトリを共有できます。これにより、ゲストはホストのファイルやフォルダをコピーせずにアクセスできます。Vagrantは、Vagrantfileのconfig.vm.synced_folderオプションを使用して共有フォルダを指定します1。Dockerは、docker runコマンドの`-v`または`--volume`フラグを使用して、ホストディレクトリをコンテナ内のデータボリュームとしてマウントします2。
VagrantとDockerはどちらも、必要なベースイメージをダウンロードできます。ベースイメージは、ゲスト環境を作成するための出発点となります。Vagrantは、Vagrantfileのconfig.vm.boxオプションを使用して、使用するベースイメージを指定します1。Dockerは、DockerfileのFROM命令を使用して、使用するベースイメージを指定します2。VagrantとDockerはどちらも、パブリックリポジトリまたはローカルソースからベースイメージをダウンロードできます。
VagrantとDockerはどちらもベースイメージに変更を加えることができます。変更とは、ゲスト環境をカスタマイズするためにベースイメージに加える修正や追加です。Vagrantはプロビジョナーを使用して、ゲストが起動した後にスクリプトやコマンドを実行します1。DockerはDockerfileの指示を使用して、ベースイメージ上でコマンドを実行し、新しいイメージを作成します2。VagrantとDockerはどちらも、ゲストが破棄された後に変更を新しいイメージに保存したり、破棄したりできます。
* Vagrant と Docker は、システム イメージの起動方法が異なります。Vagrant は、デフォルトでは VirtualBox、VMware、Hyper-V1 などのプロバイダを使用して、システム イメージを仮想マシンとして起動します。Docker は、デフォルトでは macOS、Linux、Windows のネイティブ コンテナ化機能を使用して、システム イメージをコンテナとして起動します。コンテナは一般的に仮想マシンよりも軽量で高速

ですが、セキュリティと柔軟性は劣ります。参考資料: 1: Vagrant vs. Docker | Vagrant | HashiCorp Developer 2: Vagrant vs Docker:
どちらがあなたに最適か？ どちらも良い場合もある）- Kinsta Web開発ツール

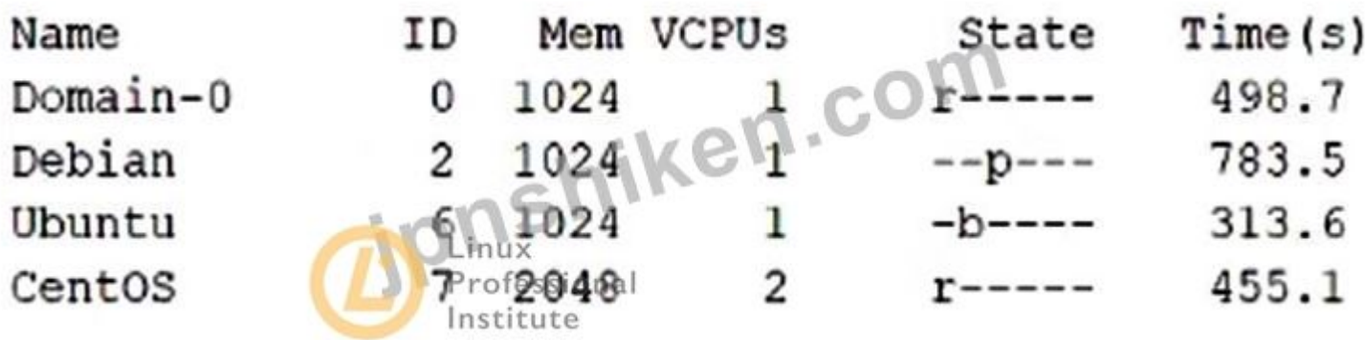
質問: **55**
完全仮想化におけるリソース管理に関して、以下の記述のうち正しいものはどれですか？ 2つ選択してください。）
A. ハイパーバイザは、各仮想マシンに定義された容量のハードウェアを提供し、仮想マシンのリソースを制限します。
B. 完全仮想化では仮想マシンに制限を設けることはできず、ホストシステムのリソースを常に先着順で割り当てます。
C. 割り当てられたハードウェア リソースを使用して、たとえば任意の数のネットワーク ソケットを作成するかどうかは、仮想マシン次第です。
D. 仮想マシン内で作成されたすべてのプロセスは、CPU および I/O の使用に関して、ホストシステム内で透過的かつ均等にスケジュールされます。
E. ハイガバイザは、プロセス数など、ゲストオペレーティングシステムの内部要素に対してきめ細かな制限を提供する場合があります。
正解: **(**正解を表示します**)**

質問: **56**
Dockerスタックを使用してDocker Swarm上でDocker Composeファイルを実行する場合、Docker Compose構成で参照されているイメージはどのようにしてSwarmノード上で利用可能になるのでしょうか？
A. docker stack はイメージをローカルでビルドし、サービスを実行する Swarm ノードにのみコピーします。
B. docker stack はイメージを Swarm マスターに渡し、マスターはイメージを他のすべての Swarm ノードに配布します。
C. docker stack は Swarm ノードにレジストリからイメージをプルするように指示しますが、イメージをレジストリにアップロードすることはありません。
D. docker stack は、ローカルの Docker キャッシュから各 Swarm ノードにイメージを転送します。
E. docker stack は、Swarm のすべてのノード上のイメージのビルド プロセスをトリガーします。
正解: **C (**コメントを発表する**)**
説明
Docker stack は、Docker Swarm クラスタ上でサービスのスタックをデプロイおよび管理できるようにするコマンドです。スタックとは、依存関係を共有し、オーケストレーションやスケーリングを一緒に実行できる、相互に関連するサービスのグループです。スタックは通常、Compose ファイルによって定義されます。Compose ファイルとは、スタックのサービス、ネットワーク、ボリューム、その他のリソースを記述する YAML ファイルです。docker stack を使用して Swarm 上で Compose ファイルを実行するには、まず Swarm クラスタを作成して初期化する必要があります。Swarm クラスタとは、Docker Engine を実

行し、単一のエンティティとして結合されたマシン (ノード) のグループです。Swarm クラスタには、クラスタの状態を維持し、サービスをオーケストレーションする責任を負う 1 つ以上のマネージャと、サービスを実行するノードである 1 つ以上のワーカーがあります。
ユーザーが Compose ファイルを指定して docker stack deploy コマンドを実行すると、コマンドはファイルを解析し、指定されたサービスを作成します。ただし、docker stack は Compose ファイルで参照されているイメージをビルドしたり、レジストリにアップロードしたりはしません。代わりに、Swarm ノードにレジストリからイメージをプルするように指示します。レジストリは、公開の Docker Hub またはプライベートレジストリのいずれかです。ユーザーは、スタックをデプロイする前に、イメージがレジストリに存在することを確認する必要があります。そうでない場合、デプロイは失敗します。ユーザーは、docker build コマンドと docker push コマンドを使用してイメージを作成し、レジストリにアップロードするか、Docker Hub や GitHub Actions などの自動ビルドサービスを使用できます。また、Compose ファイル内のイメージ名とタグがレジストリ内のものと一致すること、およびレジストリがプライベートの場合は Swarm ノードがレジストリにアクセスできることを確認する必要があります。レジストリからイメージをプルすることで、docker stack は Swarm ノードが同じ最新バージョンのイメージを持ち、イメージがクラスタ全体に効率的に分散されることを保証します。
他の選択肢は正しくありません。Docker Stack はイメージをローカルまたは Swarm ノード上でビルドせず、イメージを Swarm ノードにコピーまたは転送することもありません。また、ボトルネックと単一障害点が発生するため、イメージを Swarm マスターに渡すこともありません。Docker Stack はイメージの真の情報源としてレジストリに依存し、イメージのプルを Swarm ノードに委任します。参考資料:
* スタックをスウォームにデプロイする | Dockerドキュメント1
* docker stack deploy | Docker Docs2
* docker build | Docker Docs3
* docker push | Docker Docs4

質問: **57**
XenStoreとのやり取りには、次のうちどのツールが使用されますか？
A. ゼエンド
B. 牛
C. xstore
D. xentore-ls
E. xlストア
正解: **D** (コメントを発表する)
XenStoreは、Xenドメインで使用される共有構成および状態データベースです。Xenのドキュメントによると、xenstore-ls、xenstore-read、xenstore-writeなどのツールを使用して、XenStoreと直接やり取りします。
オプションDには軽微なタイプミス（「xentore-ls」ではなく「xenstore-ls」）がありますが、正しくドキュメント化されたユーティリティを指していることは明らかです。その他のオプションはいずれも、標準的なXenStore操作ツールを表すものではありません。
したがって、タイプミスはあるものの、これが正解です。

質問: **58**
次のxlリストの出力に関して、正しい記述はどれですか？



Name	ID	Mem	VCPUs	State	Time(s)
Domain-0	0	1024	1	r-----	498.7
Debian	2	1024	1	--p---	783.5
Ubuntu	6	1024	1	-b----	313.6
CentOS	7	2048	2	r-----	455.1

japanshiken.com
Linux Professional Institute

A. ID 2 のドメインは Para 仮想化を使用しています。
B. DebianとUbuntuはどちらも起動時にxlコマンドが必要です。
C. Ubuntu はアイドル状態か、I/O を待機中です。
D. Ubuntuの状態を実行中に変更するには、xlコマンドを使用する必要があります。
E. CentOS は最も多くの CPU 時間を消費したドメインです。
正解: (正解を表示します)

質問: **59**
Vagrantの「Vagrantfile」設定ファイルの主な目的は何ですか？
A. クラウドプロバイダーの詳細を指定する
B. 利用可能なソフトウェアパッケージの一覧を提供する
C. 開発者の氏名と連絡先情報を指定する
D. 仮想マシンのハードウェア要件を定義する
正解: (正解を表示します)

質問: **60**
Intel Extended Page Table (EPT) や AMD Rapid Virtualization Indexing (RVI) などのネストされたページテーブル拡張機能をサポートする CPU によって、どのハードウェア コンポーネントの仮想化が促進されますか？
A. ホストバスアダプタ
B. IOキャッシュ
C. ハードディスク
D. ネットワークインターフェース
E. メモリ
正解: (正解を表示します)

質問: **61**
libvirtとは何ですか？
A. 仮想化管理API。

B. Linuxディストリビューション
C. 仮想化技術
D. ハードウェア仮想化プラットフォーム
正解: **A** (コメントを発表する)
Libvirtは、仮想化プラットフォームを一貫性のある安全な方法で管理するために設計された、仮想化管理API、ツールキット、およびコマンドラインユーティリティのセットです。公式の仮想化およびコンテナ化に関するドキュメントによると、LibvirtはKVM、QEMU、Xen、VMware ESXi、Hyper-Vなどの複数のハイパーバイザとやり取りするための共通インターフェースを提供し、それらの根本的な違いを抽象化します。
Libvirt自体は仮想化技術やハイパーバイザではなく、管理者やオーケストレーションツールとハイパーバイザ間の管理レイヤーとして機能します。これにより、仮想マシンの作成、起動、停止、一時停止、移行、監視といったタスクが可能になります。virsやvirt-managerなどのツールは、仮想化操作を実行するためにLibvirtに依存しています。
libvirtの大きな強みの一つは、セキュリティと分離に重点を置いている点です。SELinuxやAppArmorといったLinuxセキュリティフレームワークと統合することで、仮想マシンに対する強制的なアクセス制御を実現します。
Libvirtは、セキュアなプロトコルを使用したリモート管理もサポートしているため、企業環境やクラウド環境に適しています。
仮想化関連のドキュメントでは、libvirtはInfrastructure as Codeと自動化の基盤となるコンポーネントとして説明されており、Ansible、OpenStack、Kubernetes（KVM経由）などのツールと併用されることが多い。その標準化されたAPIは、移植性、拡張性、ベストプラクティスへの準拠を確保しながら、仮想化管理を簡素化する。

有効的な**305-300**問題集はJPNTest.com提供され、**305-300**試験に合格することに役に立ちます！JPNTest.comは今最新**305-300**試験問題集を提供します。JPNTest.com 305-300試験問題集はもう更新されました。ここで**305-300**問題集のテストエンジンを手に入れます。最新版のアクセス、https://www.jpntest.com/shiken/305-300-mondaishu　**125**問、３０%ディスカウント、特別な割引コード: **JPNshiken**」

質問: **62**
VirtualBoxに関して、以下の記述のうち正しいものはどれですか？
A. 仮想マシンのディスクイメージを仮想化ホストのブロックデバイス上にローカルに保存できないため、専用の共有ストレージが必要です。
B. 仮想マシンを管理するためのグラフィカルユーザーインターフェースとコマンドラインツールの両方を提供します。
C. ゲストオペレーティングシステムとしてLinuxのみをサポートしており、仮想マシン内でWindowsを実行することはできません。

D. これは、最初の通常のオペレーティングシステムが起動する前に起動される特別なカーネルとして設計されたハイパーバイザーです。
E. Linux専用であり、現在実行中のLinuxカーネルのソースコードが利用可能である必要があります。
正解: **(**正解を表示します**)**

質問: **63**
cloud-initの設定ファイルの一般的なフォーマットは何ですか？
A. .xml
B. .json
C. .ini
D. .yaml
正解: **(**正解を表示します**)**
Cloud-initの設定ファイルは通常、YAML（Yet Another Markup Language）形式で記述されます。
cloud-initの公式ドキュメントでは、cloud-configfilesはYAML構文を使用することが明記されています。YAML構文は人間が読みやすく、宣言的な設定に適しています。
YAMLファイルを使用すると、管理者はユーザー、パッケージ、サービス、ネットワーク、コマンドを構造化された一貫性のある方法で定義できます。cloud-initはシェルスクリプトやMIMEマルチパートデータを処理できますが、YAMLは依然として標準かつ推奨されるフォーマットです。
したがって、正解はDです。

質問: **64**
Dockerスタックを使用してDocker Swarm上でDocker Composeファイルを実行する場合、Docker Compose構成で参照されているイメージはどのようにしてSwarmノード上で利用可能になるのでしょうか？
A. docker stack はイメージを Swarm マスターに渡し、マスターはイメージを他のすべての Swarm ノードに配布します。
B. docker stack は Swarm ノードにレジストリからイメージをプルするように指示しますが、イメージをレジストリにアップロードすることはありません。
C. docker stack は、Swarm のすべてのノード上のイメージのビルド プロセスをトリガーします。
D. docker stack は、ローカルの Docker キャッシュから各 Swarm ノードにイメージを転送します。
E. docker stack はイメージをローカルでビルドし、サービスを実行する Swarm ノードにのみコピーします。
正解: **B (**コメントを発表する**)**

質問: **65**
コンテナ仮想化は、従来の仮想化と比較してどのような主な利点がありますか？
A. 資源利用率が低い
B. 起動時間の短縮
C. ハードウェアとの互換性の向上
D. アプリケーション間の分離性の向上

正解: **B** (コメントを発表する)
コンテナ仮想化の最大の利点は、起動時間の短縮です。コンテナはホストOSのカーネルを共有するため、完全なゲストOSを起動する必要がなく、数秒または数ミリ秒で起動できます。
コンテナはリソース消費量も少ないが、ドキュメントで最もよく挙げられる主な利点は、起動の速さと拡張性であり、したがってBが正解となる。

質問: **66**
コマンドlxc-checkpointに関する以下の記述のうち、正しいものはどれですか？
A. コンテナのクローンを作成します。
B. コンテナのメモリ消費量が2倍になります。
C. 停止中のコンテナでのみ機能します。
D. コンテナの状態をファイルに書き込みます。
E. 既存のコンテナに基づいてコンテナイメージを作成します。
正解: **(**正解を表示します**)**
説明
lxc-checkpoint コマンドは、コンテナのチェックポイントと復元に使用されます。コンテナのチェックポイントとは、メモリ、プロセス、ファイルディスクリプタ、ネットワーク接続など、コンテナの状態をファイルまたはディレクトリに保存することです。コンテナの復元とは、保存された状態からコンテナを再開し、停止していなかった状態に戻すことです。コンテナのチェックポイントと復元は、ライブマイグレーション、バックアップ、デバッグ、スナップショットなど、さまざまな用途に役立ちます。lxc-checkpoint コマンドの構文は次のとおりです。
lxc-checkpoint {-n 名前} {-D パス} [-r] [-s] [-v] [-d] [-F]
選択肢は以下のとおりです。
* -n name: チェックポイントを作成または復元するコンテナの名前を指定します。
* -D path: チェックポイントデータがダンプまたは復元されるファイルまたはディレクトリへのパスを指定します。
* -r、--restore: コンテナのチェックポイントをダンプする代わりに復元します。このオプションは -s と互換性がありません。
* -s、--stop: ダンプ後にコンテナを停止します（オプション）。このオプションは -r と互換性がありません。
* -v、--verbose: criu の詳細なログ記録を有効にします。-r を指定した場合のみ利用可能です。
* -d、--daemon: コンテナをバックグラウンドで復元します (これがデフォルトです)。-r を指定した場合のみ利用可能です。
* -F、--foreground: コンテナをフォアグラウンドで復元します。-r を指定した場合のみ利用可能です。
コマンド lxc-checkpoint は、CRIU (Checkpoint/Restore In Userspace) ツールを使用してチェックポイントと復元操作を実行します。CRIU は、実行中のアプリケーション (またはその一部) を一時停止し、ファイルの集合としてハードドライブにチェックポイントを作成できるソフトウェアです。その後、これらのファイルを使用して、一時停止した時点からアプリケーションを復元して実行できます。

lxc-checkpoint コマンドに関するその他の記述は正しくありません。このコマンドはコンテナのクローンやイメージを作成するものではなく、コンテナのメモリ使用量を倍増させるものでもありません。指定されたオプションによっては、実行中のコンテナと停止中のコンテナの両方で動作します。参考資料：
* Linuxコンテナ - LXC - マニュアルページ - lxc-checkpoint.12
* lxc-checkpoint(1) - Linux マニュアルページ - man7.org3
* CRIU4

質問: **67**
LXCコンテナにデフォルトで存在するデバイスは次のうちどれですか？ （3つ選択してください。）
A. /dev/log
B. /dev/console
C. /dev/urandom
D. /dev/kmem
E. /dev/root
正解: **(**正解を表示します**)**
説明
LXC（Linux Containers）は、複数の独立したLinuxシステム（コンテナ）を同一ホスト上で実行できる軽量仮想化技術です。LXCは、名前空間、cgroups、AppArmorといったLinuxカーネルの機能を利用してコンテナを作成・管理します。各コンテナは独自のファイルシステム、ネットワークインターフェース、プロセスツリー、リソース制限を持ちます。しかし、コンテナはホストと同じカーネルとハードウェアを共有するため、完全な仮想化よりも効率的で高速です。
デフォルトでは、LXCコンテナには動作に必要な最小限のデバイスセットが用意されています。これらのデバイスは、コンテナの起動時にLXCライブラリによって作成され、コンテナの停止時に削除されます。デフォルトのデバイスは以下のとおりです。
* /dev/log: これは、ホスト上の syslog デーモンに接続する Unix ドメインソケットです。これにより、コンテナはログメッセージをホストのシステム log1 に送信できます。
* /dev/console: これは、コンテナのコンソールへのアクセスを提供するキャラクタデバイスです。通常はホストのターミナルまたはファイルに接続されます。これにより、コンテナはユーザーまたはホストのinitシステム12と対話できます。
* /dev/urandom: これは、無制限の擬似乱数を提供する文字デバイスです。暗号化、UUID生成、ハッシュ化13など、ランダム性を必要とするさまざまなアプリケーションやライブラリで使用されます。
質問に記載されているその他のデバイスは、LXCコンテナにはデフォルトでは存在しません。それらは不要であるか、許可されていないか、コンテナの名前空間またはcgroup構成でサポートされていないためです。これらのデバイスは以下のとおりです。
* /dev/kmem: これはカーネルの仮想メモリへのアクセスを提供するキャラクタデバイスです。コンテナは /proc ファイルシステムを介して自身のメモリにアクセスできるため、このデバイスは必要ありません。また、ホストのカーネルメモリを公開してセキュリティを損なうことになるため、コンテナではこのデバイスの使用は許可されていません。4

* /dev/root: これはシステムのルートデバイスを指すシンボリックリンクです。コンテナはホストとは別のルートデバイスを持たないため、このリンクはコンテナではサポートされていません。コンテナのルートファイルシステムは、ホスト5上のディレクトリ、イメージファイル、またはループデバイスからマウントされます。
参考文献：
* Linuxコンテナ - LXC - マニュアルページ - lxc.container.conf.5
* Linuxコンテナ - LXC - 入門編
* 乱数生成 - Wikipedia
* /dev/kmem - Wikipedia
* Linuxコンテナ - LXC - マニュアルページ - lxc.container.conf.5

質問: **68**
KVM仮想化ホストをセットアップする際に、次のうち不要なコンポーネントはどれですか？
A. bridgeutils
B. KVMカーネルモジュール
C. qemu
D. libvirt
E. virsh
正解: **(**正解を表示します**)**
KVMベースの仮想化ホストを構成する際には、仮想マシンを有効にして管理するために、いくつかのコアコンポーネントが必須となります。KVMおよび仮想化に関するドキュメントによると、KVMカーネルモジュールはLinuxカーネル内でハードウェア支援型仮想化をサポートするため不可欠です。QEMUはハードウェアエミュレーションを実行し、仮想マシンの実行を管理するために必要です。Libvirtは仮想化管理APIとして機能し、仮想マシンの集中管理とセキュリティを確保します。さらに、bridgeutilsはネットワークブリッジを構成するために一般的に必要であり、仮想マシンが外部ネットワークと通信できるようにします。
ただし、virshは必須ではありません。virshはlibvirtと連携して仮想マシンを管理するコマンドラインユーティリティですが、あくまで管理インターフェースであり、コア依存関係ではありません。virshをインストールしなくても、virt-manager、Ansible、OpenStack、またはカスタムAPIベースのソリューションなどの代替ツールを使用して仮想マシンを作成および管理できます。
仮想化に関するドキュメントでは、必須のバックエンドコンポーネント（KVM、QEMU、libvirt）とオプションの管理ツール（virsh）が明確に区別されています。そのため、virshは管理の利便性から広く利用され、強く推奨されていますが、機能的なKVM仮想化ホストにとって必須のコンポーネントではありません。

質問: **69**
以下のコマンドを使用して新しいDockerネットワークを作成した後：
docker network create --driver bridge isolated_nw
コンテナをネットワークに接続するために、docker createコマンドにどのパラメータを追加する必要がありますか？
A. --eth0=isolated_nw

B. --alias=isolated_nw
C. --ethernet=isolated_nw
D. --network=isolated_nw
E. --attach=isolated_nw
正解: (正解を表示します)
説明
コンテナを作成する際に、コンテナをネットワークに接続するには、引数としてネットワーク名とともに --network フラグを使用する必要があります。--network フラグは、コンテナのネットワークモードを指定します。デフォルトでは、ネットワークモードは bridge であり、コンテナはデフォルトのブリッジネットワークに接続されます。ただし、この場合 isolated_nw のようなカスタムネットワークを作成する場合は、--network フラグを使用してコンテナを明示的にそのネットワークに接続する必要があります。たとえば、web1 という名前のコンテナを作成し、それを isolated_nw ネットワークに接続するには、次のコマンドを使用します。
docker create --name web1 --network isolated_nw nginx
その他のオプションは、docker create の有効なパラメータではありません。--eth0、--ethernet、および --attach フラグは存在しません。--alias フラグは、ユーザー定義ネットワーク上のコンテナに追加のネットワークエイリアスを指定するために使用されますが、コンテナをネットワークに接続するものではありません。参照:
* docker network create | Docker ドキュメント1
* docker create | Docker ドキュメント
* ネットワークの概要 | Docker Docs2

質問: **70**
Xenゲストドメインの構成ファイルにあるオプションビルダーで、準仮想化マシンではなく完全仮想化マシンを作成するために設定する必要がある値はどれですか？
正解:
ビルダー = "hvm"
Explanation:
Xen仮想化では、ゲストドメインは準仮想化（PV）またはハードウェア支援型完全仮想化（HVM：ハードウェア仮想マシンとも呼ばれる）のいずれかを使用して作成できます。公式のXenドキュメントによると、Xenゲストドメイン構成ファイルのビルダーオプションによって、仮想マシンの作成時に使用される仮想化の種類が決まります。
完全仮想化（HVM）ゲストを作成するには、構成ファイルでビルダーオプションの値を明示的に「hvm」に設定する必要があります。
ビルダー = "hvm"
これは、Xenツールスタックに対し、Intel VT-xやAMD-Vなど、CPUが提供するハードウェア支援型仮想化機能を使用するように指示するもので、変更を加えていないゲストOSを実行できます。完全に仮想化されたゲストは物理マシンのように動作し、Xenを認識しないOSでも実行できます。

ビルダーオプションが省略されている場合、またはデフォルトの準仮想化ビルダーに設定されている場合、XenはPV対応カーネルを想定しており、そのためにはゲストOSの変更が必要となります。一方、HVMゲストはデバイスエミュレーションとハードウェア仮想化のサポートに依存しています。
仮想化ドキュメントでは、PV ゲストと HVM ゲストを明確に区別し、ビルダーを識別します
完全な仮想化に必要な構成として「hvm」を指定します。この設定は、WindowsやXen固有のカーネルを含まない標準的なLinuxディストリビューションなどのオペレーティングシステムをデプロイする際に不可欠です。
したがって、正しい値として文書化されているのは builder = "hvm" です。

質問: **71**
KubernetesのPodに関して、以下の記述のうち正しいものはどれですか？ （2つ選択してください。）
A. Pod のすべてのコンテナは同じノードで実行されます。
B. Podは常に自動的に作成され、明示的に設定することはできません。
C. Pod は、Kubernetes が実行できるワークロードの最小単位です。
D. Pod が失敗した場合、Kubernetes はデフォルトで別のノードで Pod を再起動します。
E. systemd は、Kubernetes ノード上の個々の Pod を管理するために使用されます。
正解: **A,C (**コメントを発表する**)**
説明
Kubernetes の Pod は、同じネットワークおよびストレージ リソースを共有する 1 つ以上のコンテナの集合であり、コンテナの実行方法に関する仕様です。Pod は Kubernetes が実行できるワークロードの最小単位であり、より小さな単位に分割することはできません。したがって、オプション C が正解です。Pod のすべてのコンテナは同じノードで実行されます。ノードは Kubernetes のコンピューティング ハードウェアの最小単位です。ノードは、1 つ以上の Pod をホストする物理マシンまたは仮想マシンです。したがって、オプション A も正解です。Pod は必ずしも自動的に作成されるわけではなく、明示的に構成することもできません。Pod は、YAML ファイルまたは JSON ファイルを使用したり、kubectl run や kubectl create などのコマンドを使用したりして手動で作成できます。Pod は、Deployment、ReplicaSet、StatefulSet などの上位レベルのコントローラーによって自動的に作成することもできます。したがって、オプション B は不正解です。
Pod が失敗した場合、Kubernetes はデフォルトでは別のノードで Pod を再起動しません。Pod は本質的に一時的なものであり、いつでも終了または削除できます。Pod がコントローラーによって管理されている場合、コントローラーは失敗した Pod を置き換える新しい Pod を作成しますが、同じノード上にあるとは限りません。したがって、オプション D は誤りです。systemd は Kubernetes ノード上の個々の Pod を管理するために使用されません。systemd は、docker や kubelet などのサービスを開始および停止できる Linux オペレーティングシステムのシステムおよびサービス マネージャーです。ただし、systemd は Pod と直接やり取りしません。Pod は、各ノードで実行され、Kubernetes コントロール プレーンと通信するエージェントである kubelet サービスによって管理されます。したがって、オプション E は誤りです。参照:
* ポッド | Kubernetes
* Kubernetesポッドとは何ですか？ - Red Hat

* ポッド、クラスター、コンテナの違いは何ですか？
* Kubernetes Podとは？ | VMware用語集
* Kubernetesのノード、ポッド、クラスター：主な違い - CloudZero

質問: **72**
Intel Extended Page Table (EPT) や AMD Rapid Virtualization Indexing (RVI) などのネストされたページテーブル拡張機能をサポートする CPU によって、どのハードウェア コンポーネントの仮想化が促進されますか？
A. メモリ
B. ネットワークインターフェース
C. ホストバスアダプタ
D. ハードディスク
E. IOキャッシュ
正解: (正解を表示します)
説明
Intel Extended Page Table (EPT) や AMD Rapid Virtualization Indexing (RVI) などのネストされたページテーブル拡張機能は、メモリの仮想化を容易にするハードウェア機能です。これらの機能により、CPU はソフトウェアで管理されるシャドウページテーブルを必要とせずに、ゲスト仮想アドレスをホスト物理アドレスに一度に変換できます。これにより、仮想マシンのメモリ管理のオーバーヘッドと複雑さが軽減され、パフォーマンスと分離性が向上します。ネストされたページテーブル拡張機能は、ネットワークインターフェイス、ホストバスアダプタ、ハードディスク、I/O キャッシュなどの他のハードウェアコンポーネントの仮想化には直接影響しません。
参考文献：
* 第二レベル住所翻訳 - Wikipedia
* c - 拡張ページテーブルの用途は何ですか？ - Stack Overflow
* ハイパーバイザ入門 - パート4：拡張アドレス変換を使用したアドレス変換...

質問: **73**
次のうち、実行中のLXCコンテナ内でコマンドを実行するコマンドはどれですか？
A. lxc-accach
B. lxc-batch
C. lxc-run
D. lxc-enter
E. lxc-eval
正解: (正解を表示します)
コマンド lxc-attach は、実行中の LXC コンテナ内でコマンドを実行するために使用されます。これにより、コンテナ内でプロセスを開始し、その標準入力、出力、およびエラー ストリームにアタッチできます 1。たとえば、コマンド lxc-attach -n mycontainer -- ls -lh /home は、mycontainer という名前のコンテナの /home ディレクトリ内のすべてのファイルとディレクトリを一覧表示します 1。その他のオプションは、有効な LXC コマンドではありません。コマンド lxc-batch は存在しません。コマンド lxc-run は lxc-

start のエイリアスであり、コンテナを起動するために使用されます。コンテナ内でコマンドを実行するために使用されるものではありません 2。コマンド lxc-enter も lxc-attach のエイリアスですが、非推奨であり、使用しないでください 3。コマンド lxc-eval も有効な LXC コマンドではありません。参照:
* 1: 実行中の LXC 内でコマンドを実行する - Unix & Linux Stack Exchange。
* 2: lxc-start: コンテナを起動します。- SysTutorials。
* 3: lxc-attach: 実行中のコンテナ内でプロセスを開始します。- SysTutorials。

質問: **74**
Xen 3.x において、xl コマンドの前身として使用されていたコマンドはどれですか？
正解:
xm
Explanation:
Xen 3.x では、仮想マシンの管理に使用される主要なコマンドラインツールは xm でした。Xen 仮想化に関するドキュメントによると、xm は当初の Xen 管理スタックの一部であり、Xen ドメインの作成、起動、停止、移行、監視を行うためのコマンドを提供していました。
Xenの進化に伴い、xmツールスタックは非推奨となり、xlツールスタックに置き換えられました。xlコマンドはXenハイパーバイザと直接連携し、パフォーマンスの向上、保守性の改善、複雑さの軽減を実現します。最新のXenインストールではxmはサポートされなくなりましたが、xlの前身として歴史的に重要な位置を占めています。
したがって、正解はxmです。

質問: **75**
以下のコマンドのうち、ハードウェア仮想化拡張機能を使用してQEMU仮想マシンを起動するのはどれですか？
A. qvirt -create -drive file=debian.img -cdrom debian.iso -m 1024 -boot d -driver hvm
B. vm -kvm -drive file=debian.img -cdrom debian.iso -m 1024 -boot d
C. qemu-hw -create -drive file=debian.img -cdrom debian.iso -m 1024 -boot d
D. qemu -accel kvm -drive file-debian.img -cdrom debian.iso -m 1024 -boot d
E. qvm start -vmx -drive file=debian.img -cdrom debian.iso -m 1024 -boot d
正解: (正解を表示します)
説明
ハードウェア仮想化拡張機能を使用してQEMU仮想マシンを起動するための正しいコマンドはqemuです。
-accel kvm -drive file-debian.img -cdrom debian.iso -m 1024 -boot d。このコマンドは、-accel オプションを使用して、使用するハードウェア アクセラレータを指定します。この場合は kvm です。KVM は、仮想化拡張機能 (Intel VT または AMD-V) を含む x86 ハードウェア上の Linux 用の完全な仮想化ソリューションです 1。-drive オプションは、使用するディスク イメージ ファイルを指定します。この場合は debian.img です。-cdrom オプションは、CD-ROM として使用する ISO イメージ ファイルを指定します。この場合は debian.iso です。-m オプションは、仮想マシンに割り当てるメモリ量を指定します。この場合

は 1024 MB です。-boot オプションは、ブート順序を指定します。この場合は d で、最初に CD-ROM から起動することを意味します。参考文献:
https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/virtualization_deployment_and_
https://fedoraproject.org/wiki/Virtualization

質問: **76**
以前使用した仮想マシンのクローンを作成する必要があります。クローンしたディスクイメージからは、ユーザーアカウント、シェル履歴、SSHホストキーなど、仮想マシン固有の情報をすべて削除する必要があります。これらのタスクを実行できるツールは次のうちどれですか？
A. virt-sparsi
B. vircリセット
C. virt-svspre
D. ウイルス除去
E. sysprep
F. virt-rescue
正解: **(**正解を表示します**)**

有効的な**305-300**問題集はJPNTest.com提供され、**305-300**試験に合格することに役に立ちます！JPNTest.comは今最新**305-300**試験問題集を提供します。JPNTest.com 305-300試験問題集はもう更新されました。ここで**305-300**問題集のテストエンジンを手に入れます。最新版のアクセス、https://www.jpntest.com/shiken/305-300-mondaishu　**125**問、３０%ディスカウント、特別な割引コード: **JPNshiken**」

質問: **77**
LXDコンテナ構成において有効なネットワークインターフェースタイプは次のうちどれですか？ 3つ選択してください。）
A. ブリッジ
B. IPsec
C. macvlan
D. 物理的
E. Wi-Fi
正解: **(**正解を表示します**)**

質問: **78**
Packerを使用してマシンイメージを作成する主な利点は何ですか？
A. 仮想マシンのリアルタイム監視を提供します
B. 開発環境と本番環境の一貫性を確保します

C. アプリケーションコードの開発を加速します
D. 仮想マシンの展開を簡素化します
正解: (正解を表示します)
Packerは、環境間で一貫性を確保できるという最大の利点を持つ画像自動化ツールです。
仮想化およびDevOpsに関するドキュメントによると、Packerを使用すると、管理者は単一の構成から開発、テスト、および本番環境用の同一のマシンイメージを作成できます。
Packerは、同じベースイメージとビルドプロセスを使用することで、設定のずれを解消し、環境の一貫した動作を保証します。これにより、環境の違いに起因するデプロイメントの問題が大幅に軽減され、Infrastructure as Codeのベストプラクティスがサポートされます。
Packerは仮想マシンを監視したり、アプリケーション開発を直接的に高速化したりするものではありません。間接的にデプロイメントを簡素化することはできますが、最大の利点はイメージの一貫性と再現性です。
したがって、正解はBです。

質問: **79**
LXCについて、次のうちどれが正しいですか？
A. Windowsのみに対応しています
B. Linuxカーネルは使用していません
C. コンテナランタイムです
D. 主に仮想マシンの管理に使用されます
正解: (正解を表示します)
LXC（Linux Containers）は、Linuxカーネルのネームスペースやcgroupなどの機能を利用して、軽量で隔離された環境を提供するコンテナランタイムです。コンテナ化に関するドキュメントによると、LXCは、同じカーネルを共有しながら、複数のLinuxシステム（コンテナ）を単一のホスト上で実行することを可能にします。
LXCはLinux固有のものであり、Linuxカーネルを直接使用するため、仮想マシンマネージャではありません。したがって、正解はCです。

質問: **80**
DockerfileのFROMステートメントで有効な値は次のうちどれですか？
A. ubuntu:focal
B. docker://ubuntu:focus
C. レジストリ:ubuntu:focal
D. file:/tmp/ubuntu/Dockerfile
E. http://docker.example.com/images/ubuntu-focal.iso
正解: (正解を表示します)
Dockerfile の FROM ステートメントは、後続の命令が実行されるベースイメージを指定します 1。FROM ステートメントの値は、イメージ名、タグ付きのイメージ名、またはイメージ ID のいずれかです 1。イメージ名は、リポジトリ名、またはレジストリプレフィックス付きのリポジトリ名のいずれかです 2。たとえば、ubuntu はリポジトリ名であり、docker.io/ubuntu はレジストリプレフィックス付きのリポジトリ

名です 2。タグは、イメージの特定のバージョンまたはバリアントを指定するために使用できるオプションの識別子です 1。たとえば、ubuntu:focal は、ubuntu リポジトリの focal タグが付いたイメージを参照します 2。イメージ ID は、イメージがビルドまたはプルされたときに自動的に生成される一意の識別子です 1。たとえば、sha256:
9b0dafaadb1cd1d14e4db51bd0f4c0d56b6b551b2982b2b7c637ca143ad605d2 は画像 ID3 です。
したがって、指定されたオプションの中で FROM ステートメントで有効な値は ubuntu:focal のみです。これはタグ付きのイメージ名です。その他のオプションは無効です。理由は以下のとおりです。
* docker://ubuntu:focal は有効なイメージ名形式ではありません。docker:// プレフィックスはトランスポート プロトコルを指定するために使用され、レジストリ プレフィックスではありません。4.
* registry:ubuntu:focal は有効なイメージ名形式ではありません。レジストリのプレフィックスは、汎用的な用語ではなく、有効なホスト名または IP アドレスである必要があります。
* file:/tmp/ubuntu/Dockerfile は有効なイメージ名形式ではありません。file: プレフィックスは、イメージ名ではなく、ローカルファイルパスを指定するために使用されます5。
* http://docker.example.com/images/ubuntu-focal.iso
有効な画像ファイル名の形式ではありません。
の
http:// プレフィックスは、画像名ではなく、Web URL を指定するために使用されます。
5.
参考文献：
1 :Dockerfileリファレンス｜Dockerドキュメント
2: docker - dockerfile で FROM ステートメントを使用する - Stack Overflow
3: Dockerイメージからイメージidを取得する方法 - Stack Overflow
4: skopeo - Docker Registry v2 API ツール - Linux マニュアルページ (1)
5 :ローカルのDockerfileからDockerイメージを構築する方法は？ - Stack Overflow

質問**: 81**
IaaSクラウドにおいて、オペレーティングシステムとソフトウェアを搭載した新しいコンピューティングインスタンスをプロビジョニングする一般的な方法は何ですか？
A. 各新規インスタンスはLinuxディストリビューションのインストールメディアに接続され、SSH経由でログインすることでインストーラーにアクセスできます。
B. 各新規インスタンスは、オペレーティングシステム、ソフトウェア、および特定の目的のためのデフォルト設定を含むイメージファイルに基づいて作成されます。
C. 新しいインスタンスはそれぞれ、現在実行中の別のインスタンスのクローンであり、元のインスタンスのすべてのソフトウェア、データ、および状態を含みます。
D. 新しいインスタンスはそれぞれ、プロビジョニングを開始したコンピュータとVPN経由で接続され、そのマシンからPXEブートを試みます。
E. 各新規インスタンスには、仮想CDから実行される最小限のライブシステムが含まれており、管理者はこれを基にターゲットオペレーティングシステムを展開します。
正解**: (**正解を表示します**)**
説明

IaaSクラウドでは、新しいコンピューティングインスタンスをプロビジョニングする最も一般的な方法は、オペレーティングシステムとソフトウェアがプリインストールされたイメージファイルを使用することです。このイメージファイルは、マシンイメージ、仮想アプライアンス、またはテンプレートとも呼ばれます。イメージファイルは、Webサーバー、データベースサーバー、開発環境など、特定の目的に合わせてカスタマイズできます。イメージファイルは、クラウドプロバイダーまたはユーザーがアクセスできるリポジトリまたはライブラリに保存できます。新しいインスタンスが要求されると、クラウドプロバイダーはイメージファイルを仮想ディスクにコピーし、インスタンスにアタッチします。インスタンスは仮想ディスクから起動し、イメージファイルからオペレーティングシステムとソフトウェアを実行します。この方法は、新しいインスタンスごとにオペレーティングシステムとソフトウェアを最初からインストールするよりも高速で効率的です。また、同じイメージファイルを使用する複数のインスタンス間で一貫性と信頼性を確保します。参考文献：
* LPI仮想化およびコンテナ化試験の目標、トピック305.1：仮想化の概念と理論、目標 :マシンイメージとテンプレートの概念を説明する
* LPI仮想化およびコンテナ化学習ガイド、第1章：仮想化の概念と理論、セクション :マシンイメージとテンプレート
* LPI LPIC-3 305 認定試験のサンプル問題と練習問題、質問 10: IaaS クラウドでは、オペレーティングシステムとソフトウェアを使用して新しいコンピューティング インスタンスをプロビジョニングする一般的な方法は何ですか?

質問: **82**
acgroupdirectory内のどのファイルに、このcgroupに属するプロセスの一覧が含まれていますか？
A. pids
B. メンバー
C. プロシージャ
D. 樽
E. 被験者
正解: **(**正解を表示します**)**
説明
cgroup ディレクトリ内のファイル procs には、この cgroup に属するプロセスの一覧が格納されています。ファイル内の各行には、cgroup のメンバーであるプロセスの PID が表示されます。プロセスを cgroup に移動するには、その PID を cgroup の procs ファイルに書き込みます。たとえば、PID 24982 のプロセスを cgroup cg1 に移動するには、次のコマンドを使用します。echo 24982 > /sys/fs/cgroup/cg1/procs1。ファイル procs は、cgroup に属するスレッドの一覧であるファイル tasks とは異なります。ファイル procs を使用すると、スレッド グループ内のすべてのスレッドを一度に移動できますが、ファイル tasks を使用すると、個々のスレッドを移動できます。2 参照:
* cgroupsの作成と整理 cgroup2 - GitHub Pages
* コントロールグループ - Linuxカーネルのドキュメント

質問: **83**

KVMドメインにおいて、libvirtによって制限できるリソースは次のうちどれですか？ 2つ選択してください。）
A. CPUライムの量
B. 利用可能なメモリのサイズ
C. ドメイン内で許可されているファイルシステム
D. 実行中のプロセス数
E. 利用可能なファイル数
正解: **(**正解を表示します**)**
Libvirtは、KVM、Xen、LXCなどのさまざまな仮想化技術を管理するための共通APIを提供するツールキットです。Libvirtを使用すると、仮想マシン（ドメインとも呼ばれます）のCPU、メモリ、ディスク、ネットワーク、その他のリソースなど、さまざまな側面を構成および制御できます。LibvirtでKVMドメインに対して制限できるリソースには、次のようなものがあります。
* CPU時間：Libvirtでは、ドメインが使用できる仮想CPU（vCPU）の数、CPUモード、モデル、トポロジ、チューニングパラメータを指定できます。また、CPUシェア、クォータ、期間を設定して、ドメインが消費できるCPU時間の相対的または絶対的な量を制御することもできます。さらに、vCPUを物理CPUまたはNUMAノードに固定して、パフォーマンスと分離性を向上させることもできます。これらの設定は、ドメインXMLファイルの<cpu>要素と<cputune>要素12で構成できます。
* 使用可能なメモリのサイズ: Libvirt では、ドメインが使用できるメモリの量、メモリのバックアップ、チューニング、および NUMA ノードのパラメータを指定できます。また、メモリのハード リミットとソフト リミット、スワップのハード リミット、および最小保証を設定して、ドメインのメモリ割り当てと再利用ポリシーを制御することもできます。これらの設定は、ドメイン XML ファイルの <memory>、<memoryBacking>、および <memtune> 要素で構成できます 13。
質問に記載されているその他のリソースは、KVM ドメインの場合、libvirt によって直接制限されるものではありません。ドメインで許可されるファイルシステムは、ドメインに接続されているディスクおよびファイルシステム デバイスによって決定され、これらはドメイン XML ファイルの <disk> および <filesystem> 要素で構成できます 14。実行中のプロセス数と使用可能なファイル数は、ドメインのオペレーティングシステムとファイルシステムによって決定され、libvirt によって制御されるものではありません。
:
libvirt: ドメインXML形式
CPU割り当て
メモリ割り当て
ハードディスク、フロッピーディスク、CD-ROM

質問: **84**
Vagrantの「Vagrantfile」設定ファイルの主な目的は何ですか？
A. 仮想マシンのハードウェア要件を定義する
B. 利用可能なソフトウェアパッケージの一覧を提供する
C. 開発者の名前と連絡先情報を指定する
D. クラウドプロバイダーの詳細を指定する

正解: (正解を表示します)
Vagrantfileは、Vagrantが仮想マシン環境を定義および制御するために使用するコア構成ファイルです。Vagrantのドキュメントによると、その主な目的は、メモリサイズ、CPU数、ネットワーク、同期フォルダ、プロビジョニング手順などのハードウェア関連設定を含む、仮想マシンの構成を記述することです。Vagrantfile を使用すると、開発者は仮想マシンの特性を再現性と移植性の高い方法で宣言でき、チーム間で一貫した開発環境を確保できます。プロバイダーやプロビジョニングツールを参照することもできますが、その基本的な役割は、仮想マシンの作成方法と構成方法を定義することです。
オプションBとCはVagrantの目的とは無関係であり、オプションDは範囲が狭すぎる。Vagrantfileの使用はクラウドプロバイダーに限定されないからだ。
したがって、正解はAです。

質問: **85**
準仮想化におけるXenの役割は何ですか？
A. 上記のいずれでもない
B. パフォーマンス向上のため、ゲストOSの変更が必要です
C. 完全仮想化ゲストのみをサポートします。
D. ハードウェアを完全に仮想化します
正解: **B** (コメントを発表する)
準仮想化では、ゲストオペレーティングシステムは仮想化環境で動作していることを認識し、ハイパーバイザと効率的に連携するように変更されます。Xenのドキュメントによると、Xenは元々準仮想化に特化したハイパーバイザとして設計されており、パフォーマンス向上のためにゲストオペレーティングシステムの変更が必要でした。
これらの変更により、ゲストOSは機密性の高いハードウェア命令をハイパーコールに置き換えることが可能になり、ハードウェアエミュレーションに伴うオーバーヘッドが削減されます。その結果、初期の完全仮想化技術と比較して、CPU、メモリ、およびI/Oのパフォーマンスが向上します。
Xenは完全仮想化（HVM）ゲストをサポートしていますが、それは準仮想化におけるXenの役割ではありません。したがって、選択肢BがXenの役割を正しく説明しています。

質問: **86**
以下のコンポーネントのうち、Dockerアーキテクチャにおいて不可欠なものはどれですか？（該当するものをすべて選択してください）
A. Dockerクライアント
B. Docker Compose
C. ドッカーデーモン
D. ドッカーレジストリ
正解: (正解を表示します)
Dockerアーキテクチャは、コンテナの作成と管理を可能にするいくつかのコアコンポーネントで構成されています。
Dockerのドキュメントによると、Dockerクライアント、Dockerデーモン、およびDockerレジストリは必須コンポーネントです。

Dockerクライアントは、デーモンにコマンドを送信するユーザー向けのコマンドラインインターフェースです。Dockerデーモン（dockerd）は、イメージの構築、コンテナの実行、ネットワークとストレージの管理など、多くの処理を実行します。Dockerレジストリはコンテナイメージを保存し、イメージの配布と再利用を可能にします。
Docker Composeは、複数のコンテナで構成されるアプリケーションを定義および実行するために使用されるオプションのオーケストレーションツールですが、必須のアーキテクチャコンポーネントではありません。
したがって、正解はA、C、Dです。

有効的な**305-300**問題集はJPNTest.com提供され、**305-300**試験に合格することに役に立ちます！JPNTest.comは今最新**305-300**試験問題集を提供します。JPNTest.com 305-300試験問題集はもう更新されました。ここで**305-300**問題集のテストエンジンを手に入れます。最新版のアクセス、https://www.jpntest.com/shiken/305-300-mondaishu　**125**問、３０%ディスカウント、特別な割引コード: **JPNshiken**」