**Vendor:** LPI

**Exam Code:** 305-300

**Exam Name:** LPIC-3: Virtualization and Containerization - Exam 305, version 3.0

**Version:** DEMO

**QUESTION 1**
Which of the following network interface types are valid in an LXD container configuration? (Choose three.)

A. ipsec
B. macvlan
C. bridged
D. physical
E. wifi

**Answer:** BCD
**Explanation:**
LXD supports the following network interface types in an LXD container configuration:

macvlan: Creates a virtual interface on the host with a unique MAC address and attaches it to an existing physical interface. This allows the container to have direct access to the physical network, but prevents communication with the host and other containers on the same host.
bridged: Connects the container to an existing bridge interface on the host. This allows the container to communicate with the host and other containers on the same bridge, as well as the external network if the bridge is connected to a physical interface.
physical: Passes an existing physical interface on the host to the container. This allows the container to have exclusive access to the physical network, but removes the interface from the host.

**QUESTION 2**
What happens when the following command is executed twice in succession?

```
docker run -tid -v data:/data debian bash
```

A. The container resulting from the second invocation can only read the content of /data/ and cannot change it.
B. Each container is equipped with its own independent data volume, available at /data/ in the respective container.
C. Both containers share the contents of the data volume, have full permissions to alter its content and mutually see their respective changes.
D. The original content of the container image data is available in both containers, although changes stay local within each container.
E. The second command invocation fails with an error stating that the volume data is already associated with a running container.

**Answer:** C
**Explanation:**
The command docker run -tid -v data:/data debian bash creates and runs a new container from the debian image, with an interactive terminal and a detached mode, and mounts a named volume data at /data in the container. If the volume data does not exist, it is created automatically. If the command is executed twice in succession, two containers are created and run, each with its own terminal and process ID, but they share the same volume data. This means that both containers can access, modify, and see the contents of the data volume, and any changes made by one container are reflected in the other container. Therefore, the statement C is true and the correct answer. The statements A, B, D, and E are false and incorrect, as they do not describe the behavior of the command or the volume correctly.

**QUESTION 3**
What is the purpose of the kubelet service in Kubernetes?

A. Provide a command line interface to manage Kubernetes.
B. Build a container image as specified in a Dockerfile.
C. Manage permissions of users when interacting with the Kubernetes API.
D. Run containers on the worker nodes according to the Kubernetes configuration.
E. Store and replicate Kubernetes configuration data.

**Answer:** D
**Explanation:**
The purpose of the kubelet service in Kubernetes is to run containers on the worker nodes according to the Kubernetes configuration. The kubelet is an agent or program that runs on each node and communicates with the Kubernetes control plane. It receives a set of PodSpecs that describe the desired state of the pods that should be running on the node, and ensures that the containers described in those PodSpecs are running and healthy. The kubelet also reports the status of the node and the pods back to the control plane. The kubelet does not manage containers that were not created by Kubernetes.

**QUESTION 4**
If docker stack is to be used to run a Docker Compose file on a Docker Swarm, how are the images referenced in the Docker Compose configuration made available on the Swarm nodes?

A. docker stack builds the images locally and copies them to only those Swarm nodes which run the service.
B. docker stack passes the images to the Swarm master which distributes the images to all other Swarm nodes.
C. docker stack instructs the Swarm nodes to pull the images from a registry, although it does not upload the images to the registry.
D. docker stack transfers the image from its local Docker cache to each Swarm node.
E. docker stack triggers the build process for the images on all nodes of the Swarm.

**Answer:** C
**Explanation:**
Docker stack is a command that allows users to deploy and manage a stack of services on a Docker Swarm cluster. A stack is a group of interrelated services that share dependencies and can be orchestrated and scaled together. A stack is typically defined by a Compose file, which is a YAML file that describes the services, networks, volumes, and other resources of the stack. To use docker stack to run a Compose file on a Swarm, the user must first create and initialize a Swarm cluster, which is a group of machines (nodes) that are running the Docker Engine and are joined into a single entity. The Swarm cluster has one or more managers, which are responsible for maintaining the cluster state and orchestrating the services, and one or more workers, which are the nodes that run the services.

When the user runs docker stack deploy with a Compose file, the command parses the file and creates the services as specified. However, docker stack does not build or upload the images referenced in the Compose file to any registry. Instead, it instructs the Swarm nodes to pull the images from a registry, which can be the public Docker Hub or a private registry. The user must ensure that the images are available in the registry before deploying the stack, otherwise the deployment will fail. The user can use docker build and docker push commands to create and upload the images to the registry, or use an automated build service such as Docker Hub or GitHub Actions. The user must also make sure that the image names and tags in the Compose file match the ones in the registry, and that the Swarm nodes have access to the registry if it is

private. By pulling the images from a registry, docker stack ensures that the Swarm nodes have the same and latest version of the images, and that the images are distributed across the cluster in an efficient way.

**QUESTION 5**
Which of the following commands executes a command in a running LXC container?

A. lxc-accach
B. lxc-batch
C. lxc-run
D. lxc-enter
E. lxc-eval

**Answer:** A
**Explanation:**
The command lxc-attach is used to execute a command in a running LXC container. It allows the user to start a process inside the container and attach to its standard input, output, and error streams. For example, the command lxc-attach -n mycontainer -- ls -lh /home will list all the files and directories in the /home directory of the container named mycontainer.

**QUESTION 6**
Which of the following statements in a Dockerfile leads to a container which outputs hello world? (Choose two.)

A. ENTRYPOINT "echo Hello World"
B. ENTRYPOINT [ "echo hello world" ]
C. ENTRYPOINT [ "echo", "hello", "world" ]
D. ENTRYPOINT echo Hello World
E. ENTRYPOINT "echo", "Hello", "World*

**Answer:** BC
**Explanation:**
The ENTRYPOINT instruction in a Dockerfile specifies the default command to run when a container is started from the image. The ENTRYPOINT instruction can be written in two forms: exec form and shell form. The exec form uses a JSON array to specify the command and its arguments, such as [ "executable", "param1", "param2" ]. The shell form uses a single string to specify the command and its arguments, such as "executable param1 param2". The shell form is converted to the exec form by adding /bin/sh -c to the beginning of the command. Therefore, the following statements in a Dockerfile are equivalent and will lead to a container that outputs hello world:

ENTRYPOINT [ "echo hello world" ] ENTRYPOINT [ "/bin/sh", "-c", "echo hello world" ] ENTRYPOINT "echo hello world" ENTRYPOINT [ "echo", "hello", "world" ] ENTRYPOINT [ "/bin/sh", "-c", "echo", "hello", "world" ] ENTRYPOINT "echo hello world"

**QUESTION 7**
Which of the following statements are true regarding a Pod in Kubernetes? (Choose two.)

A. All containers of a Pod run on the same node.
B. Pods are always created automatically and cannot be explicitly configured.
C. A Pod is the smallest unit of workload Kubernetes can run.

D. When a Pod fails, Kubernetes restarts the Pod on another node by default.
E. systemd is used to manage individual Pods on the Kubernetes nodes.

**Answer:** AC
**Explanation:**
A Pod in Kubernetes is a collection of one or more containers that share the same network and storage resources, and a specification for how to run the containers. A Pod is the smallest unit of workload Kubernetes can run, meaning that it cannot be divided into smaller units. Therefore, option C is correct. All containers of a Pod run on the same node, which is the smallest unit of computing hardware in Kubernetes. A node is a physical or virtual machine that hosts one or more Pods. Therefore, option A is also correct. Pods are not always created automatically and cannot be explicitly configured. Pods can be created manually using YAML or JSON files, or using commands like kubectl run or kubectl create. Pods can also be created automatically by higher-level controllers, such as Deployment, ReplicaSet, or StatefulSet. Therefore, option B is incorrect. When a Pod fails, Kubernetes does not restart the Pod on another node by default. Pods are ephemeral by nature, meaning that they can be terminated or deleted at any time. If a Pod is managed by a controller, the controller will create a new Pod to replace the failed one, but it may not be on the same node. Therefore, option D is incorrect. systemd is not used to manage individual Pods on the Kubernetes nodes. systemd is a system and service manager for Linux operating systems that can start and stop services, such as docker or kubelet. However, systemd does not interact with Pods directly. Pods are managed by the kubelet service, which is an agent that runs on each node and communicates with the Kubernetes control plane. Therefore, option E is incorrect.

**QUESTION 8**
Which of the following values would be valid in the FROM statement in a Dockerfile?

A. ubuntu:focal
B. docker://ubuntu: focal
C. registry:ubuntu:focal
D. file:/tmp/ubuntu/Dockerfile
E. http://docker.example.com/images/ubuntu-focal.iso

**Answer:** A
**Explanation:**
The FROM statement in a Dockerfile specifies the base image from which the subsequent instructions are executed. The value of the FROM statement can be either an image name, an image name with a tag, or an image ID. The image name can be either a repository name or a repository name with a registry prefix. For example, ubuntu is a repository name, and docker.io/ubuntu is a repository name with a registry prefix. The tag is an optional identifier that can be used to specify a particular version or variant of an image. For example, ubuntu:focal refers to the image with the focal tag in the ubuntu repository. The image ID is a unique identifier that is automatically generated when an image is built or pulled. For example, sha256:9b0dafaadb1cd1d14e4db51bd0f4c0d56b6b551b2982b2b7c637ca143ad605d2 is an image ID.

Therefore, the only valid value in the FROM statement among the given options is ubuntu:focal, which is an image name with a tag.

**QUESTION 9**
Which CPU flag indicates the hardware virtualization capability on an AMD CPU?

A. HVM

B. VIRT
C. SVM
D. PVM
E. VMX

**Answer:** C
**Explanation:**
The CPU flag that indicates the hardware virtualization capability on an AMD CPU is SVM. SVM stands for Secure Virtual Machine, and it is a feature of AMD processors that enables the CPU to run virtual machines with hardware assistance. SVM is also known as AMD-V, which is AMD's brand name for its virtualization technology. SVM allows the CPU to support a hypervisor, which is a software layer that creates and manages virtual machines. A hypervisor can run multiple virtual machines on a single physical machine, each with its own operating system and applications. SVM improves the performance and security of virtual machines by allowing the CPU to directly execute privileged instructions and handle memory access, instead of relying on software emulation or binary translation. SVM also provides nested virtualization, which is the ability to run a virtual machine inside another virtual machine. To use SVM, the CPU must support it and the BIOS must enable it. The user can check if the CPU supports SVM by looking for the svm flag in the /proc/cpuinfo file or by using the lscpu command. The user can also use the virt-host-validate command to verify if the CPU and the BIOS are properly configured for hardware virtualization.

**QUESTION 10**
What is the purpose of a .dockerignore file?

A. It lists files existing in a Docker image which should be excluded when building a derivative image.
B. It specifies files that Docker does not submit to the Docker daemon when building a Docker image
C. It exists in the root file system of containers that should ignore volumes and ports provided by Docker.
D. It must be placed in the top level directory of volumes that Docker should never attach automatically to a container
E. It specifies which parts of a Dockerfile should be ignored when building a Docker image.

**Answer:** B
**Explanation:**
The purpose of a .dockerignore file is to specify files that Docker does not submit to the Docker daemon when building a Docker image. A .dockerignore file is a text file that contains a list of files or directories that should be excluded from the build context, which is the set of files and folders that are available for use in a Dockerfile. By using a .dockerignore file, you can avoid sending files or directories that are large, contain sensitive information, or are irrelevant to the Docker image to the daemon, which can improve the efficiency and security of the build process.

**QUESTION 11**
Which file in a cgroup directory contains the list of processes belonging to this cgroup?

A. pids
B. members
C. procs
D. casks
E. subjects

**Answer:** C
**Explanation:**
The file procs in a cgroup directory contains the list of processes belonging to this cgroup. Each line in the file shows the PID of a process that is a member of the cgroup. A process can be moved to a cgroup by writing its PID into the cgroup's procs file. For example, to move the process with PID 24982 to the cgroup cg1, the following command can be used: echo 24982 > /sys/fs/cgroup/cg1/procs. The file procs is different from the file tasks, which lists the threads belonging to the cgroup. The file procs can be used to move all threads in a thread group at once, while the file tasks can be used to move individual threads.

**QUESTION 12**
What kind of virtualization is implemented by LXC?

A. System containers
B. Application containers
C. Hardware containers
D. CPU emulation
E. Paravirtualization

**Answer:** A
**Explanation:**
LXC implements system containers, which are a type of operating-system-level virtualization. System containers allow running multiple isolated Linux systems on a single Linux control host, using a single Linux kernel. System containers share the same kernel with the host and each other, but have their own file system, libraries, and processes. System containers are different from application containers, which are designed to run a single application or service in an isolated environment. Application containers are usually smaller and more portable than system containers, but also more dependent on the host kernel and libraries. Hardware containers, CPU emulation, and paravirtualization are not related to LXC, as they are different kinds of virtualization methods that involve hardware abstraction, instruction translation, or modification of the guest operating system.

**QUESTION 13**
Which of the following statements is true regarding networking with libvirt?

A. Libvirt's network functionality is limited to connecting virtual machines to a physical network interface of the host system.
B. Libvirt assiqns the same MAC address to all virtual machines and isolates their network interfaces at the link layer.
C. Libvirt networks appear, by default, as standard Linux bridges in the host system.
D. Libvirt requires a dedicated network interface that may not be used by the host system.
E. Libvirt supports exactly one virtual network and connects all virtual machines to it.

**Answer:** C
**Explanation:**
Libvirt supports creating and managing various types of virtual networks that can be used to connect virtual machines to each other or to the external network. One of the common types of virtual networks is the NAT-based network, which uses network address translation (NAT) to allow virtual machines to access the outside world through the host's network interface. By default, libvirt creates a NAT-based network called `default' when it is installed and started. This network appears as a standard Linux bridge device on the host system, named virbr0. The bridge

device has an IP address of 192.168.122.1/24 and acts as a gateway and a DHCP server for the virtual machines connected to it. The bridge device also has iptables rules to forward and masquerade the traffic from and to the virtual machines. The virtual machines connected to the `default' network have their own IP addresses in the 192.168.122.0/24 range and their own MAC addresses generated by libvirt. The virtual machines can communicate with each other, with the host, and with the external network through the bridge device and the NAT mechanism.

**QUESTION 14**
Which of the following are true regarding the CPU of a QEMU virtual machine? (Choose two.)

A. The CPU architecture of a QEMU virtual machine is independent of the host system's architecture.
B. Each QEMU virtual machine can only have one CPU with one core.
C. For each QEMU virtual machine, one dedicated physical CPU core must be reserved.
D. QEMU uses the concept of virtual CPUs to map the virtual machines to physical CPUs.
E. QEMU virtual machines support multiple virtual CPUs in order to run SMP systems.

**Answer:** AE
**Explanation:**
The CPU architecture of a QEMU virtual machine is independent of the host system's architecture. QEMU can emulate many CPU architectures, including x86, ARM, Alpha, and SPARC, regardless of the host system's architecture. This allows QEMU to run guest operating systems that are not compatible with the host system's hardware. Therefore, option A is correct. QEMU virtual machines support multiple virtual CPUs in order to run SMP systems. QEMU uses the concept of virtual CPUs (vCPUs) to map the virtual machines to physical CPUs. Each vCPU is a thread that runs on a physical CPU core. QEMU allows the user to specify the number of vCPUs and the CPU model for each virtual machine. QEMU can run SMP systems with multiple vCPUs, as well as single-processor systems with one vCPU.

**QUESTION 15**
Which of the following statements about the command lxc-checkpoint is correct?

A. It creates a clone of a container.
B. It doubles the memory consumption of the container.
C. It only works on stopped containers.
D. It writes the status of the container to a file.
E. It creates a container image based on an existing container.

**Answer:** D
**Explanation:**
The command lxc-checkpoint is used to checkpoint and restore containers. Checkpointing a container means saving the state of the container, including its memory, processes, file descriptors, and network connections, to a file or a directory. Restoring a container means resuming the container from the saved state, as if it was never stopped. Checkpointing and restoring containers can be useful for various purposes, such as live migration, backup, debugging, or snapshotting. The command lxc- checkpoint has the following syntax:

```
lxc-checkpoint {-n name} {-D path} [-r] [-s] [-v] [-d] [-F]
```

The options are:

-n name: Specify the name of the container to checkpoint or restore.

-D path: Specify the path to the file or directory where the checkpoint data is dumped or restored.
-r, --restore: Restore the checkpoint for the container, instead of dumping it. This option is incompatible with -s.
-s, --stop: Optionally stop the container after dumping. This option is incompatible with -r.
-v, --verbose: Enable verbose criu logging. Only available when providing -r.
-d, --daemon: Restore the container in the background (this is the default). Only available when providing -r.
-F, --foreground: Restore the container in the foreground. Only available when providing -r.

The command lxc-checkpoint uses the CRIU (Checkpoint/Restore In Userspace) tool to perform the checkpoint and restore operations. CRIU is a software that can freeze a running application (or part of it) and checkpoint it to a hard drive as a collection of files. It can then use the files to restore and run the application from the point it was frozen at.

**QUESTION 16**
The command virsh vol-list vms returns the following error:

```
error: failed to get pool 'vms'
error: Storage pool not found: no storage pool with matching name 'vms
'
```

Given that the directory /vms exists, which of the following commands resolves this issue?

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

**Answer:** D
**Explanation:**
The command virsh pool-create-as vms dir --target /vms creates and starts a transient storage pool named vms of type dir with the target directory /vms. This command resolves the issue of the storage pool not found error, as it makes the existing directory /vms visible to libvirt as a storage pool. The other commands are invalid because:

dd if=/dev/zero of=/vms bs=1 count=0 flags=name:vms is not a valid command syntax. The dd command does not take a flags argument, and the output file /vms should be a regular file, not a directory.
libvirt-poolctl new --name=/vms --type=dir --path=/vms is not a valid command name. There is no such command as libvirt-poolctl in the libvirt package.
qemu-img pool vms:/vms is not a valid command syntax. The qemu-img command does not have a pool subcommand, and the vms:/vms argument is not a valid image specification.
touch /vms/.libvirtpool is not a valid command to create a storage pool. The touch command only creates an empty file, and the .libvirtpool file is not recognized by libvirt as a storage pool configuration file.

# Thank You for Trying Our Product

## Passleader Certification Exam Features:

★ More than **99,900** Satisfied Customers Worldwide.

★ Average **99.9%** Success Rate.

★ **Free Update** to match latest and real exam scenarios.

★ **Instant Download** Access! No Setup required.

★ Questions & Answers are downloadable in **PDF** format and **VCE** test engine format.

★ Multi-Platform capabilities - **Windows, Laptop, Mac, Android, iPhone, iPod, iPad**.

★ **100%** Guaranteed Success or **100%** Money Back Guarantee.

★ **Fast**, helpful support **24x7**.

View list of all certification exams: http://www.passleader.com/all-products.html

**10% Discount Coupon Code:** **ASTR14**