Huawei Cloud EulerOS (HCE)

Best Practices

Issue	01
Date	2025-10-09

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: https://www.huaweicloud.com/intl/en-us/

Contents

1 Deploying MySQL

Introduction

MySQL is an open-source relational database management system. Like other relational databases, MySQL stores data in tables consisting of rows and columns. Users can define, manipulate, control, and query data using a structured query language (usually called SQL). This section uses MySQL 8.0.42 as an example to describe how to install and deploy MySQL in HCE.

Preparations

- Prepare two ECSs and assign a public IP address or an EIP to each of them.
- Ensure that inbound security group rules allow traffic to flow to the ECS over ports 22 and 3306.

Prerequisites

A yum repository has been configured. For details, see **Configuring Repositories and Installing Software for HCE**.

Procedure

Step 1 Install the MySQL server and client.

1. Install the MySQL server and client.
   ```
   dnf install -y mysql-server mysql-common mysql
   ```
2. Check the MySQL version.
   ```
   mysql -V
   ```
   If information similar to the following is displayed, the installation is successful:
   ```
   mysql  Ver 8.0.42 for Linux on x86_64 (Source distribution)
   ```

Step 2 Configure MySQL.

1. Start the MySQL service.
   ```
   systemctl start mysqld
   ```
2. Check the MySQL status.
   ```
   systemctl status mysqld
   ```
   If **active (running)** is displayed, MySQL is started.

NOTE

To set mysqld to automatically enable upon system boot, run the following command: systemctl enable mysqld

3. Perform security configuration for MySQL.

```
mysql_secure_installation
```

Customize options as needed.

```
Securing the MySQL server deployment.

Connecting to MySQL using a blank password.

VALIDATE PASSWORD COMPONENT can be used to test passwords
and improve security. It checks the strength of password
and allows the users to set only those passwords which are
secure enough. Would you like to setup VALIDATE PASSWORD component?

Press y|Y for Yes, any other key for No: Y // Choose whether to set the password component. Y is recommended.

There are three levels of password validation policy:

LOW    Length >= 8
MEDIUM Length >= 8, numeric, mixed case, and special characters
STRONG Length >= 8, numeric, mixed case, special characters and dictionary file

Please enter 0 = LOW, 1 = MEDIUM and 2 = STRONG: 2 // Select the password security policy. 2 is recommended.
Please set the password for root here.

New password: // Set the password based on the password strength you selected.

Re-enter new password: // Enter the new password again.

Estimated strength of the password: 100
Do you wish to continue with the password provided?(Press y|Y for Yes, any other key for No) : y
By default, a MySQL installation has an anonymous user,
allowing anyone to log in to MySQL without having to have
a user account created for them. This is intended only for
testing, and to make the installation go a bit smoother.
You should remove them before moving into a production
environment.

Remove anonymous users? (Press y|Y for Yes, any other key for No) : y // Choose whether to remove anonymous users. y is recommended.
Success.


Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.

Disallow root login remotely? (Press y|Y for Yes, any other key for No) : y // Choose whether to disallow remote login of root. y is recommended.
Success.

By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.


Remove test database and access to it? (Press y|Y for Yes, any other key for No) : y // Choose whether to remove the test database. y is recommended.
 - Dropping test database...
Success.

 - Removing privileges on test database...
```

```
Success.

Reloading the privilege tables will ensure that all changes
made so far will take effect immediately.

Reload privilege tables now? (Press y|Y for Yes, any other key for No) : y // Choose whether to reload privilege tables. y is recommended.
Success.

All done!
```

Step 3 Remotely connect to the database.

1. Connect to the database.
   ```
   mysql -uroot -p
   ```
   Enter the password of the **root** user.
2. Create a test account and password.
   ```
   create user 'test'@'%' identified by 'xxxxxx';
   ```
   xxxxxx is the password.
3. Grant all database permissions to the test account.
   ```
   grant all privileges on *.* to 'test'@'%';
   ```
4. Update the permissions.
   ```
   flush privileges;
   ```
5. Exit the database.
   ```
   exit
   ```
6. Install the MySQL client on another ECS.
   ```
   dnf install -y mysql-common mysql
   ```
7. Connect to the database from the client.
   ```
   mysql -h <MySQL server IP> -utest -p
   ```
   Alternatively, use the Navicat or Visual Studio Code plug-in for remote connections.

----End

CAUTION

If the database connection failed, check whether the firewall is disabled or whether the MySQL port is enabled. The default port is 3306. The password and permission settings are used for tests only. Do not use them in a service environment unless absolutely necessary.

2 Installing Docker

Introduction

Docker is an open-source application container engine that features portability, scalability, high security, and manageability. Developers can package an application and its dependencies into a portable container and run the container on a Linux server to build, deploy, and manage the application more efficiently. This section describes how to install Docker in HCE.

Preparations

Prepare an ECS.

Prerequisites

A yum repository has been configured. For details, see **Configuring Repositories and Installing Software for HCE**.

Procedure

Step 1 Install Docker.

1. Install Docker.
   ```
   dnf install docker
   ```

Step 2 Verify that Docker is installed.

1. Start Docker.
   ```
   systemctl start docker
   ```
2. Check the Docker version. If a version is displayed, Docker is installed successfully.
   ```
   docker -v
   ```

Figure 2-1 Checking whether Docker is installed successfully

```
[root@localhost ~]# docker -v
Docker version 18.09.0, build 472bb7b
[root@localhost ~]#
```

----End

> **CAUTION**
>
> Docker 18.09.0 is installed. To install a later version, see **Docker Documentation**.

Using Docker

1. Manage the Docker daemon.
   - Run the daemon.
     ```
     systemctl start docker
     ```
   - Stop the daemon.
     ```
     systemctl stop docker
     ```
   - Restart the daemon.
     ```
     systemctl restart docker
     ```
2. Manage images. The **nginx** image is used as an example. You need to prepare a local image file **nginx_latest.tar**.
   - Import the image.
     ```
     docker load -i <nginx_latest.tar file path>
     ```
   - Change the image tag.
     ```
     # Change the tag of the nginx image from latest to v1.
     docker tag nginx:latest nginx:v1
     ```
   - Check existing images.
     ```
     docker images
     ```
   - Delete the image.
     ```
     docker rmi nginx:v1
     ```
3. Manage containers.
   - Start a container.
     ```
     # Create and start a container from an image. The image ID can be obtained by running docker images.
     docker run -it <image-ID> /bin/bash
     ```
   - Exit a container.
     ```
     exit
     ```
   - Start a container in the background.
     ```
     docker run -itd <image-ID>
     ```
   - Check all running containers.
     ```
     docker ps
     ```
   - Enter a container running in the background. You can run **docker ps** to obtain the container ID.
     ```
     docker exec -it <Container ID> /bin/bash
     ```
   - Create an image from the container. The container ID can be obtained by running **docker ps**.
     ```
     docker commit <Container ID> [<Repository name>[:<Tag>]]
     ```
     Example:
     ```
     docker commit 135****9f757 nginx:v1
     ```


3 Setting Up an FTP Site

Introduction

File Transfer Protocol (FTP) is a standard protocol for transferring files between computers over a network. It uses the client-server architecture and allows users to upload, download, rename, delete, and view files and directories from a remote server. FTP can work in active or passive mode:

- Active mode: The server initiatively connects to a port of the client to transmit data.
- Passive mode: The client initiatively connects to an open port of the server. This mode is applicable to network environments with firewalls.

The following types of users can log in to FTP servers:

1. Anonymous users
   - Users can log in as **anonymous** or **ftp** without a password.
   - This type of user is usually used to download public resources, for example, downloading open-source software from mirror sites.
   - Such users are insecure and are only used to access certain directories such as **/pub**. Generally, they only have the read-only permission.
2. Local users
   - They are local users in Linux.
   - Local users are usually used to share resources on an intranet or in a development and test environment.
   - The username and password are from **/etc/passwd** and **/etc/shadow**. Local users have the same permissions as system users, which means certain operation risks.
3. Virtual users
   - A virtual user does not exist in the system. They are stored in an independent account database. Generally, virtual users are mapped to a host user (for example, **vsftpd**) for access control.
   - GDBM or libdb is usually used for authentication of virtual users.
   - Virtual users do not depend on system users, so they are more secure. Additionally, they have fine-grained permissions.

vsftpd (very secure FTP daemon) is an FTP server for Unix-like systems, including Linux. This section describes how to install and configure vsftpd in HCE. Local users and virtual users will be involved.

Preparations

- Prepare two ECSs and assign a public IP address or an EIP to each of them. Use one ECS as the FTP server and the other as the client.
- Configure inbound security group rules to allow traffic to the FTP server over port 21.

Prerequisites

A yum repository has been configured. For details, see **Configuring Repositories and Installing Software for HCE**.

Installing vsftpd

Step 1 Install vsftpd.

```
dnf install vsftpd
```

Step 2 Start vsftpd.

```
systemctl start vsftpd
```

Step 3 Check the vsftpd status.

```
systemctl status vsftpd
```

If **active (running)** is displayed, vsftpd is started, as shown in **Figure 3-1**.

Figure 3-1 Example: vsftpd status

```
[root@hce2 ~]# systemctl status vsftpd
● vsftpd.service - Vsftpd ftp daemon
     Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; disabled; vendor preset: disabled)
     Active: active (running) since Tue 2023-10-10 15:55:11 CST; 1s ago
    Process: 21033 ExecStart=/usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf (code=exited, status=0/SUCCESS)
   Main PID: 21034 (vsftpd)
      Tasks: 1 (limit: 198730)
     Memory: 408.0K
     CGroup: /system.slice/vsftpd.service
             └─21034 /usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf
```

NOTE

Enable vsftpd to automatically start up upon system boot if needed.

```
systemctl enable vsftpd
```

----End

Configuring the FTP Server to Use Local User Authentication

Step 1 Add a local user for the FTP service.

1. Create a local user for the FTP service:
   ```
   adduser ftpuser
   ```
2. Set a password for the new user.
   ```
   passwd ftpuser
   ```

Step 2 Configure the vsftpd service.

1. Create a directory and file for the FTP service (the following are examples only).
```
mkdir -p /data/ftp/
touch /data/ftp/test.txt
```
2. Set the owner of the directory to the new user.
```
chown -R ftpuser:ftpuser /data/ftp/
```
3. Open the **/etc/vsftpd/vsftpd.conf** file and set the following parameters:
```
# Listen on IPv4 sockets.
listen=YES
# Determine whether to listen on IPv6 sockets.
listen_ipv6=NO
```
Append other parameters to the configuration file. Set **pasv_address** to an available public IP address or EIP.
```
# Set the root directory of the local user.
local_root=/data/ftp
# Restrict all users to their home directories.
chroot_local_user=YES
# Enable the passive mode.
pasv_enable=YES
pasv_address=<public-IP address-of-the-FTP-server>
# Determine whether to allow users to access other directories.
chroot_list_enable=NO
allow_writeable_chroot=YES
# If chroot_list_enable is set to YES, you need to set chroot_list_file to a file that contains the users who can access other directories.
# chroot_list_file=/etc/vsftpd/chroot_list
# Set the port range that can be used in passive mode. Use a high port range to improve the security of accessing the FTP server.
# Specify the lower limit of the port range. In this example, the value is 5000. You can change it based on the site requirements.
pasv_min_port=5000
# Specify the higher limit of the port range. In this example, the value is 5010. You can change it based on the site requirements.
pasv_max_port=5010
```

NOTE

Retain the default values for other parameters.

CAUTION

The **/etc/vsftpd/chroot_list** file must be created regardless of whether **chroot_list_enable** is configured.

4. Restart vsftpd.
```
systemctl restart vsftpd
```

Step 3 Verify the configuration.

1. Check the port used by vsftpd.
```
netstat -natp | grep vsftpd
```
By default, port 21 is used.
2. Configure security group inbound rules to enable port 21 and all ports between **pasv_min_port** and **pasv_max_port** specified in **/etc/vsftpd/vsftpd.conf** (ports 5000 to 5010 are used as an example).
3. Run the following command on the client ECS to install the FTP client:
```
dnf install -y ftp
```
4. Run the following command on the client to access the FTP service:
```
ftp <public-IP address-of-the-FTP-server>
```


5. Enter the username and password as prompted.
6. After the login is successful, run **ls**. If the **test.txt** file is displayed, the configuration is successful.

Figure 3-2 Verification example

```
[root@localhost ~]# ftp 90.101.69.202
Connected to 90.101.69.202 (90.101.69.202).
220 (vsFTPd 3.0.3)
Name (90.101.69.202:root): ftpuser
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
227 Entering Passive Mode (90,101,69,202,19,143).
150 Here comes the directory listing.
-rw-r--r--    1 1001     1001            0 Jun 11 11:31 test.txt
226 Directory send OK.
ftp>
```

----End

Configuring the FTP Server to Use Virtual User Authentication

Step 1 Create a host account for virtual users and disable shell login.

```
useradd vsftpd -s /bin/false
```

Virtual users will be mapped to the host account **vsftpd**. This account is used to access system resources but cannot be used to log in to the system.

Step 2 Create a GDBM database.

```
gdbmtool /etc/vsftpd/login.pag store ftpuser 123456
```

- **ftpuser**: virtual username (example)
- **123456**: password (example)
- **login.pag**: file for storing user passwords

Step 3 Configure Pluggable Authentication Modules (PAM). Open the **/etc/pam.d/vsftpd** file, delete the existing content, and add the following content:

```
auth    required    pam_userdb.so db=/etc/vsftpd/login
account required    pam_userdb.so db=/etc/vsftpd/login
```

NOTE

db=/etc/vsftpd/login indicates **login.pag** without the file name extension.

Step 4 Edit the vsftpd configuration file **/etc/vsftpd/vsftpd.conf** to add the following content:

```
guest_enable=YES
guest_username=vsftpd
user_config_dir=/etc/vsftpd/user_conf
# Enable the passive mode.
pasv_enable=YES
pasv_address=<public-IP address-of-the-FTP-server>
# Determine whether to allow users to access other directories.
chroot_list_enable=NO
allow_writeable_chroot=YES
```



```
# Set the port range that can be used in passive mode. Use a high port range to improve the security of accessing the FTP server.
# Specify the lower limit of the port range. In this example, the value is 5000. You can change it based on the site requirements.
pasv_min_port=5000
# Specify the higher limit of the port range. In this example, the value is 5010. You can change it based on the site requirements.
pasv_max_port=5010
```

Step 5 Configure permissions of the virtual user.

1. Create a directory for configuring user permissions.
   ```
   mkdir -p /etc/vsftpd/user_conf
   ```
2. Write the following content to the **/etc/vsftpd/user_conf/ftpuser** file:
   ```
   # Allow write operations (upload, rename, and delete).
   write_enable=YES
   # Allow local system users (including virtual users) to log in to the FTP server.
   local_enable=YES
   # Set the permission mask used by a local user to create a file or directory.
   local_umask=077
   # Set the permission mask used by an anonymous user to create a file or directory.
   anon_umask=022
   # Set the root directory for virtual users or local users. These users can only access this directory and its subdirectories.
   local_root=/home/vsftpd/ftp
   # Allow anonymous users to upload files.
   anon_upload_enable=YES
   # Allow anonymous users to create directories.
   anon_mkdir_write_enable=YES
   # Allow anonymous users to perform other write operations, such as renaming and deleting files.
   anon_other_write_enable=yes
   ```

   NOTE

   For vsftp, virtual users are mapped to a local user (host user), but their permissions are controlled in the same way as anonymous users. So, some anonymous user settings are also needed to enable the write permission of virtual users.

Step 6 Prepare the FTP root directory and assign permissions.

```
mkdir -p /home/vsftpd/ftp/pub
chmod 777 /home/vsftpd/ftp/pub
```

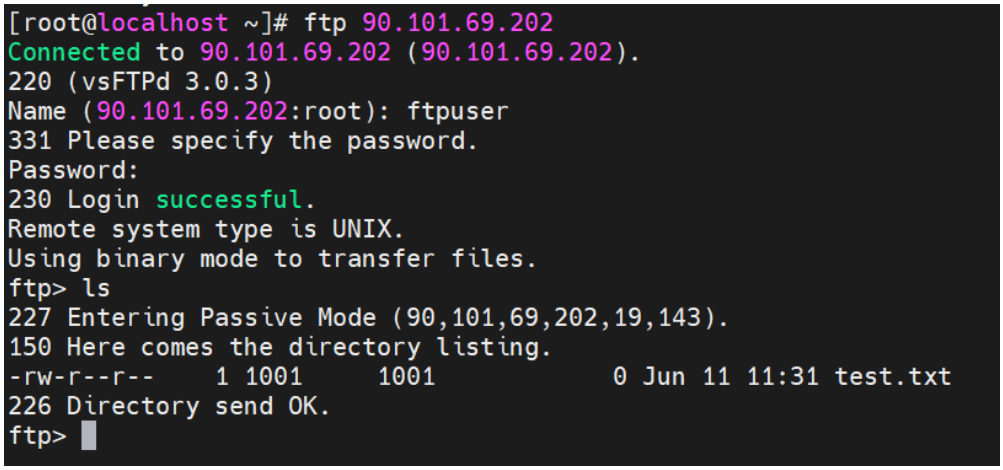
Step 7 Restart the vsftpd service.

```
systemctl restart vsftpd
```

Step 8 Verify the configuration.

1. Check the port used by vsftpd.
   ```
   netstat -natp | grep vsftpd
   ```
   By default, port 21 is used.
2. Configure security group inbound rules to enable port 21 and ports 5000 to 5010.
3. Run the following command on the client ECS to install the FTP client:
   ```
   dnf install -y ftp
   ```
4. Run the following command on the client to access the FTP service:
   ```
   ftp <public-IP address-of-the-FTP-server>
   ```
5. Enter the username and password as prompted.
6. After the login is successful, run **ls**. If the **test.txt** file is displayed, the configuration is successful.

Figure 3-3 Verification example



----End

4 Deploying the SFTP Service

Introduction

Secure File Transfer Protocol (SFTP) is a file transfer protocol that leverages a set of utilities that provide secure access to a remote computer to deliver secure communications. It is considered by many to be the optimal method for secure file transfer. It leverages SSH and provides security and identity authentication functions of SSH. This section describes how to deploy the SFTP service in HCE.

Preparations

- Prepare an ECS and assign a public IP address or EIP to the ECS.
- Ensure that inbound security group rules allow traffic to flow to the ECS over port 22.

Procedure

1. Run the following command to create a user group named **sftp**:
   ```
   groupadd sftp
   ```
2. Run the following command to create a user for accessing the SFTP service:
   ```
   useradd -g sftp -s /bin/false usftp
   ```
3. Run the following command to set a password for the new user:
   ```
   passwd usftp
   ```
4. Run the following command to create a home directory for the **sftp** user group:
   ```
   mkdir -p /data/sftp/usftp
   ```
5. Run the following command to change the login directory of the **usftp** user:
   ```
   usermod -d /data/sftp/usftp usftp
   ```
6. Open the **/etc/ssh/sshd_config** file and set the following parameters:
   ```
   # Configure Subsystem as follows:
   Subsystem sftp internal-sftp

   # Append the following content to the end of the file:
   Match user usftp # Match the usftp user.
   AllowTcpForwarding no # TCP forwarding is not allowed.
   X11Forwarding no # X11 forwarding is not allowed.
   # Use chroot to specify /data/sftp/%u as the root directory of the user. %u indicates the user name.
   ChrootDirectory /data/sftp/usftp
   ForceCommand internal-sftp # Forcibly execute internal-sftp.
   ```

   Restart the sshd service.


```
systemctl restart sshd
```

7. Run the following command to create a test file:
   ```
   touch /data/sftp/usftp/test.txt
   ```
8. Run the following commands to set the directory permission:
   ```
   chown -R root:sftp /data/sftp/usftp
   chmod 755 /data/sftp/usftp
   ```
9. On another server, run the following command to connect to the SFTP service:
   ```
   sftp usftp@<public-IP-address-of-the-SFTP-service>
   ```
10. Enter the password and run **ls** to check the test file.
    ```
    sftp> ls
    test.txt
    ```


5 Deploying PostgreSQL

Introduction

PostgreSQL is an open-source, highly-stable database system that provides SQL features, such as foreign keys, subqueries, triggers, and user-defined data types and functions. It further enhances the SQL language and provides some fine-grained capabilities to scale data workloads. It is mainly used in mobile, network, geospatial, and analysis applications and is known as the most advanced open-source database in the industry. This section describes how to deploy PostgreSQL in HCE.

Preparations

- Prepare two ECSs and assign a public IP address or an EIP to each of them. One ECS is used as the primary node, and the other is used as the standby node.
- Ensure that inbound security group rules allow traffic to flow to the ECSs over port 5432.

NOTICE

5432 is a high-risk port. Before using it, ensure that you have fully understood the potential risks.

Prerequisites

A yum repository has been configured. For details, see **Configuring Repositories and Installing Software for HCE**.

Procedure

Step 1 Configure the primary PostgreSQL node.

1. Run the following command to install the server, client, and related components:

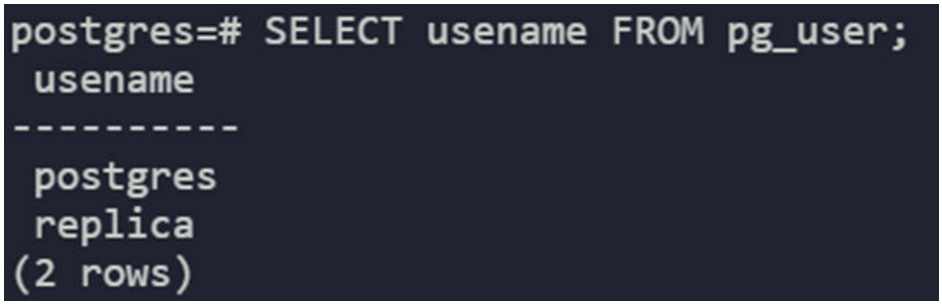
```
dnf install -y postgresql postgresql-contrib postgresql-server
```


2. Run the following command to initialize the database:
```
postgresql-setup --initdb --unit postgresql
```
3. Run the following commands in sequence to start PostgreSQL and check the PostgreSQL status:
```
systemctl start postgresql
systemctl status postgresql
```
If **active (running)** is displayed, PostgreSQL is started.

NOTE

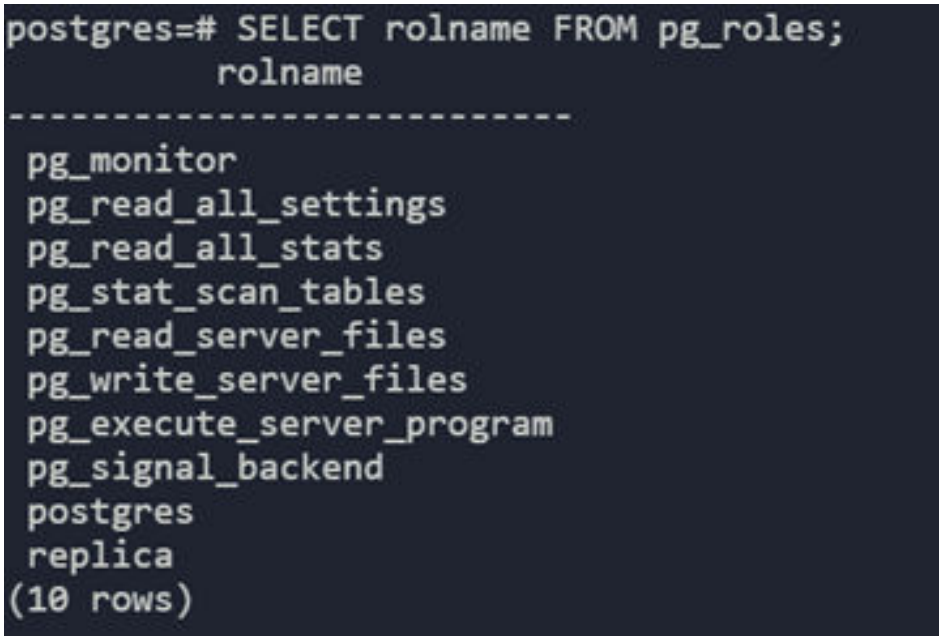
To set PostgreSQL to automatically enable upon system boot, run the following commands:
```
systemctl enable postgresql
```
4. Log in to the database as user **postgres**:
```
su - postgres
```
5. Access the psql terminal.
```
psql
```
6. Create an account and set the password and permissions.
```
CREATE ROLE replica login replication ENCRYPTED PASSWORD 'xxxxxx';
```
replica is the account name. **replica** has the login and replication permissions. *xxxxxx* is the password.
7. Check the new account.
```
SELECT usename FROM pg_user;
```
If the result shown in **Figure 5-1** is displayed, the account is created successfully.

Figure 5-1 New account



8. Check the new permissions.
```
SELECT rolname FROM pg_roles;
```
If the result shown in **Figure 5-2** is displayed, permissions are created successfully.

Figure 5-2 New permissions



9. Exit the psql terminal.
   ```
   \q
   ```
10. Exit user **postgres**.
    ```
    exit
    ```
11. Edit the **/var/lib/pgsql/data/pg_hba.conf** file to configure the **replica** user whitelist.

    Find **IPv4 local connections** and add the following information:

    ```
    host    all             all             <IPv4 address range of the secondary node>          md5
    host    replication     replica         <IPv4 address range of the secondary node>          md5
    ```

    Enter the IPv4 address range of the secondary node.
12. Open the **/var/lib/pgsql/data/postgresql.conf** file and configure the following parameters:
    ```
    listen_addresses = '*' # Private IP address listened on
    max_connections = 100 # Maximum number of connections. The value of max_connections of the secondary node must be greater than that of the primary node.
    wal_level = hot_standby # Enable the hot standby mode.
    synchronous_commit = on # Enable synchronous replication.
    max_wal_senders = 32 # Maximum number of synchronization processes
    wal_sender_timeout = 60s # Timeout for the streaming replication host to send data
    ```
13. Run the following command to restart PostgreSQL:
    ```
    systemctl restart postgresql
    ```

Step 2 Configure the secondary PostgreSQL node.

1. Run the following command to install the server, client, and related components on the secondary node:
   ```
   dnf install -y postgresql postgresql-contrib postgresql-server
   ```
2. Create a backup directory on the secondary node.
   ```
   pg_basebackup -D /var/lib/pgsql/data -h <IP address of the primary node> -p 5432 -U replica -X stream -P -R
   ```

   The options are described as follows:

   -D: backup directory

   -h: IP address or server name of the primary database

-p: port of the primary database

-U: username for connecting to the database

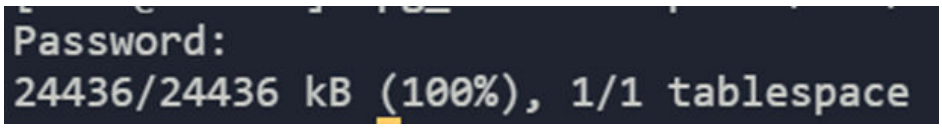
-X: WAL file replication mode (**stream** indicates streaming replication)

-P: progress

-R: automatically writes replication settings

If information shown in **Figure 5-3** is displayed, the backup is successful.

Figure 5-3 Successful backup



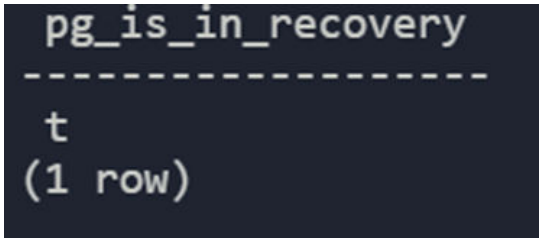
The **standby.signal** file is generated in the **/var/lib/pgsql/data** directory, and information about the primary database connections is automatically written into the **postgresql.auto.conf** file.

3. Run the following command to change the user group that the directory belongs to:
```
chown -R postgres.postgres /var/lib/pgsql/data
```
4. Run the following command to restart PostgreSQL:
```
systemctl restart postgresql
```

Step 3 Check whether the configuration is successful.

1. Run the following command on the secondary node to check whether the node is configured successfully:
```
sudo -u postgres psql -c "SELECT pg_is_in_recovery()"
```
If information shown in **Figure 5-4** is displayed, the secondary node is configured successfully.

Figure 5-4 Successful configuration of the secondary node



2. Run the following command on the primary node to obtain the secondary node information:
```
sudo -u postgres psql -x -c "SELECT * FROM pg_stat_replication" -d postgres
```
If information shown in **Figure 5-5** is displayed, the configuration is successful.

Figure 5-5 Successful configuration

```
-[ RECORD 1 ]----+------------------------------
pid              | 6119
usesysid         | 16384
usename          | replica
application_name | walreceiver
client_addr      | [redacted]
client_hostname  |
client_port      | 59124
backend_start    | 2023-10-11 11:12:25.566176+08
backend_xmin     |
state            | streaming
sent_lsn         | 0/7000148
write_lsn        | 0/7000148
flush_lsn        | 0/7000148
replay_lsn       | 0/7000148
write_lag        |
flush_lag        |
replay_lag       |
sync_priority    | 0
sync_state       | async
reply_time       | 2023-10-11 10:43:33.056944+08
```

----End

CAUTION

The password and permission settings are used for tests only. Do not use them in a service environment unless absolutely necessary.

6 Deploying Redis

Introduction

Remote Dictionary Server (Redis) is an open-source, in-memory, distributed, and durable key-value pair storage database written in C. Redis is a storage system with rich functions and applies to various scenarios, including caching, session storage, rankings, and real-time analysis. It is widely used and has active community support. This section describes how to deploy Redis in HCE.

Preparations

- Prepare two ECSs and assign a public IP address or an EIP to each ECS. One ECS is used as the primary Redis node, and the other is used as the secondary Redis node.
- Ensure that inbound security group rules allow traffic to flow to the ECSs over port 6379.

NOTICE

6379 is a high-risk port. Before using it, ensure that you have fully understood the potential risks.

Prerequisites

A yum repository has been configured. For details, see **Configuring Repositories and Installing Software for HCE**.

Procedure

Step 1 Install and configure Redis.

1. Run the following command on the two ECSs to install Redis:
   ```
   dnf install redis
   ```
2. Run the following command on the two ECSs to start Redis:
   ```
   systemctl start redis
   ```


> **NOTE**
>
> To enable Redis to start upon system boot, run the following command:
>
> ```
> systemctl enable redis
> ```

3. Run the following command to check the Redis status:
   ```
   systemctl status redis
   ```
   If **active (running)** is displayed, Redis is started.
4. On the primary node, edit the **/etc/redis.conf** file and configure the following attributes:
   ```
   bind <primary node IP> # Enter the IP address of the primary node.
   requirepass ******* # Set a password.
   ```
5. On the secondary node, edit the **/etc/redis.conf** file and configure the following attributes:
   ```
   bind <IP address of the secondary node> # Enter the IP address of the secondary node.
   requirepass *******                # Set a password.
   slaveof <IP address of the primary node> <Port of the primary node>
   masterauth <Password of the master node>
   ```
6. Run the following command on the two ECSs to restart Redis:
   ```
   systemctl restart redis
   ```

Step 2 Verify Redis.

1. Run the following command on the master node to connect to Redis:
   ```
   redis-cli -h <IP address of the primary node>
   auth <Password>
   ```
2. Run the following command to check the node information:
   ```
   127.0.0.1:6379> info replication
   # Replication
   role:master
   connected_slaves:1
   slave0:ip=x.x.x.x,port=6379,state=online,offset=4382,lag=0
   master_replid:5d68ccf7722f461cc5f004c7e96fd7c506990508
   master_replid2:0000000000000000000000000000000000000000
   master_repl_offset:4382
   second_repl_offset:-1
   repl_backlog_active:1
   repl_backlog_size:1048576
   repl_backlog_first_byte_offset:1
   repl_backlog_histlen:4382
   127.0.0.1:6379>
   ```

----End

> **CAUTION**
>
> The preceding configuration and Redis deployment architecture are used only for tests. Exercise caution when using them in the service environment.

7 Installing Tomcat

Introduction

Apache Tomcat (or simply Tomcat) is an open-source web server and servlet container developed by the Apache Software Foundation. This section describes how to deploy Tomcat in HCE.

Preparations

- Prepare an ECS and assign a public IP address or EIP to the ECS.
- Ensure that inbound security group rules allow traffic to flow to the ECS over port 8080.

Prerequisites

A yum repository has been configured. For details, see **Configuring Repositories and Installing Software for HCE**.

Procedure

Step 1 Install Tomcat.

1. Run the following command to install Java:
   ```
   dnf install java-1.8.0-openjdk
   ```
2. Run the following command to check whether the installation is successful:
   ```
   java -version
   ```
3. Run the following command to install Tomcat:
   ```
   dnf install tomcat
   ```
   The Tomcat is installed in the **/usr/share/tomcat** directory.

Step 2 Configure Tomcat.

1. Find the Java path in the **/usr/lib/jvm/** directory. If the **java-1.8.0-openjdk-*xxx*** (*xxx* indicates a version) file exists in the directory, the JRE path of Java is **/usr/lib/jvm/java-1.8.0-openjdk-*xxx*/jre**.
2. Edit the **/etc/profile** file to configure environment variables. Add the following content to the file (**JAVA_HOME** is the JRE path obtained in **Step 2.1**):
   ```
   JAVA_HOME={JRE path}
   PATH=$PATH:$JAVA_HOME/bin
   ```



```
CLASSPATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
export JAVA_HOME CLASSPATH PATH
```

3. Run the following command to activate the preceding environment variables:
   ```
   source /etc/profile
   ```
4. Clear the **/usr/share/tomcat/conf/server.xml** file and paste the following content:
   ```
   <?xml version="1.0" encoding="UTF-8"?> <Server port="8006" shutdown="SHUTDOWN"> <Listener
   className="org.apache.catalina.core.JreMemoryLeakPreventionListener"/> <Listener
   className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener"/> <Listener
   className="org.apache.catalina.core.ThreadLocalLeakPreventionListener"/> <Listener
   className="org.apache.catalina.core.AprLifecycleListener"/> <GlobalNamingResources> <Resource
   name="UserDatabase" auth="Container"  type="org.apache.catalina.UserDatabase"
   description="User database that can be updated and saved"
   factory="org.apache.catalina.users.MemoryUserDatabaseFactory"  pathname="conf/tomcat-
   users.xml"/> </GlobalNamingResources> <Service name="Catalina"> <Connector port="8080"
   protocol="HTTP/1.1"  connectionTimeout="20000"  redirectPort="8443"  maxThreads="1000"
   minSpareThreads="20"  acceptCount="1000"  maxHttpHeaderSize="65536"  debug="0"
   disableUploadTimeout="true"  useBodyEncodingForURI="true"  enableLookups="false"
   URIEncoding="UTF-8"/> <Engine name="Catalina" defaultHost="localhost"> <Realm
   className="org.apache.catalina.realm.LockOutRealm"> <Realm
   className="org.apache.catalina.realm.UserDatabaseRealm"   resourceName="UserDatabase"/> </
   Realm> <Host name="localhost" appBase="/data/wwwroot/default" unpackWARs="true"
   autoDeploy="true"> <Context path="" docBase="/data/wwwroot/default" debug="0"
   reloadable="false" crossContext="true"/> <Valve
   className="org.apache.catalina.valves.AccessLogValve" directory="logs"
   prefix="localhost_access_log." suffix=".txt" pattern="%h %l %u %t %s %b" /> </Host> </Engine> </
   Service> </Server>
   ```

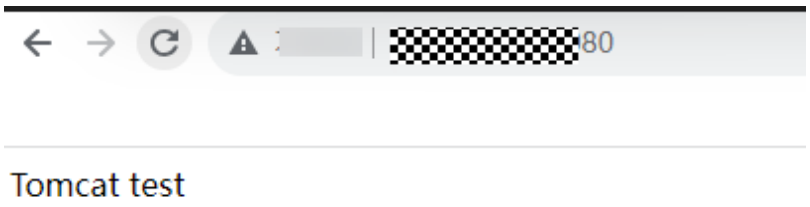
   Save the settings and exit. You can customize the preceding configuration as required.
5. Run the following command to create the directory specified by **appbase** and **docbase** in the preceding configuration:
   ```
   mkdir -p /data/wwwroot/default
   ```
6. Run the following command to set **tomcat** as the owner of the preceding directory:
   ```
   chown -R tomcat.tomcat /data/wwwroot/
   ```
7. Create the **/usr/share/tomcat/bin/setenv.sh** file and enter the following information to set JVM memory parameters:
   ```
   JAVA_OPTS='-Djava.security.egd=file:/dev/./urandom -server -Xms256m -Xmx496m -
   Dfile.encoding=UTF-8'
   ```
8. Run the following command to start Tomcat:
   ```
   systemctl start tomcat
   ```
9. Run the following command to check the Tomcat status:
   ```
   systemctl status tomcat
   ```

   If **active (running)** is displayed, Tomcat is started.

Step 3 Check whether Tomcat is installed.

1. Run the following command to create a test page:
   ```
   echo Tomcat test > /data/wwwroot/default/index.jsp
   ```
2. Enter **http://***<public-IP-address-of-the-Tomcat-service>***:8080** in the address box of a browser. If the following page is displayed, the installation is successful.

Figure 7-1 Successful installation



----End

CAUTION

The preceding configuration is used only for tests. Exercise caution when using the configuration in the service environment.

8 Installing Kafka

Introduction

Kafka is distributed pub/sub messaging middleware with high throughput, data persistence, horizontal scalability, and stream data processing. Common applications include log collection, data transmission, online/offline analysis, and real-time monitoring. This section describes how to deploy Kafka in HCE.

Preparations

- Prepare an ECS and assign a public IP address or EIP to the ECS.
- Ensure that inbound security group rules allow traffic to flow to the ECS over port 9092.

Prerequisites

A yum repository has been configured. For details, see **Configuring Repositories and Installing Software for HCE**.

Procedure

Step 1 Install Kafka.

Run the following command to install Kafka:

```
dnf install kafka
```

After the command is executed, Kafka is installed in the **/opt/kafka** directory.

Step 2 Configure Kafka.

1. Open the **/opt/kafka/config/server.properties** file and modify the following attributes:
   ```
   listeners=PLAINTEXT://<private-IP-address>:9092
   advertised.listeners=PLAINTEXT://<public-IP-address>:9092
   ```
2. Create the **/lib/systemd/system/zookeeper.service** file and enter the following information:
   ```
   [Unit]
   Description=Zookeeper service
   After=network.target

   [Service]
   ```



```
Type=simple
Environment="PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
User=root
Group=root
ExecStart=/opt/kafka/bin/zookeeper-server-start.sh /opt/kafka/config/zookeeper.properties
ExecStop=/opt/kafka/bin/zookeeper-server-stop.sh
Restart=on-failure
SuccessExitStatus=143

[Install]
WantedBy=multi-user.target
```

3. Create the **/lib/systemd/system/kafka.service** file and enter the following information:

```
[Unit]
Description=Apache Kafka server (broker)
After=network.target  zookeeper.service

[Service]
Type=simple
Environment="PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
User=root
Group=root
ExecStart=/opt/kafka/bin/kafka-server-start.sh /opt/kafka/config/server.properties
ExecStop=/opt/kafka/bin/kafka-server-stop.sh
Restart=on-failure
SuccessExitStatus=143

[Install]
WantedBy=multi-user.target
```

4. Run the following commands in sequence to start Kafka and ZooKeeper:

```
systemctl daemon-reload
systemctl start zookeeper
systemctl start kafka
```

Step 3 Verify Kafka.

1. Run the following command to create a topic:

```
 /opt/kafka/bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test
```

2. Run the following command to view the new topic:

```
/opt/kafka/bin/kafka-topics.sh --list --zookeeper localhost:2181
```

   If **test** is displayed, Kafka is deployed.

----End

CAUTION

The preceding configuration is used only for tests. Exercise caution when using the configuration in the service environment.

9 Installing GNOME

Introduction

GNU Network Object Model Environment (GNOME) is an open-source desktop environment. It provides a functional, easy-to-use, and user-friendly graphical user interface (GUI) for Linux and other Unix-like operating systems. This section describes how to install GNOME in HCE.

Preparations

Prepare an ECS.

Prerequisites

A yum repository has been configured. For details, see **Configuring Repositories and Installing Software for HCE**.

Procedure

Step 1 Install GNOME.

Run the following command to install GNOME:

```
dnf install gnome-initial-setup gnome-terminal
```

Step 2 Set GNOME to start upon system boot.

1. Run the following command to enable GNOME to start upon system boot:
   ```
   systemctl set-default graphical.target
   ```
2. Run the following command to restart GNOME:
   ```
   reboot
   ```

----End

10 Installing Apache

Introduction

Apache HTTP Server (Apache for short) is free open-source cross-platform web server software developed by the Apache Software Foundation. It can run in most operating systems and is widely used because of its high security. As one of the most popular web servers, Apache supports FastCGI and SSL and can integrate with Perl. This section describes how to deploy Apache in HCE.

Preparations

- Prepare an ECS and assign a public IP address or EIP to the ECS.
- Ensure that inbound security group rules allow traffic to flow to the ECS over port 80.

Prerequisites

A yum repository has been configured. For details, see **Configuring Repositories and Installing Software for HCE**.

Procedure

Step 1 Install Apache.

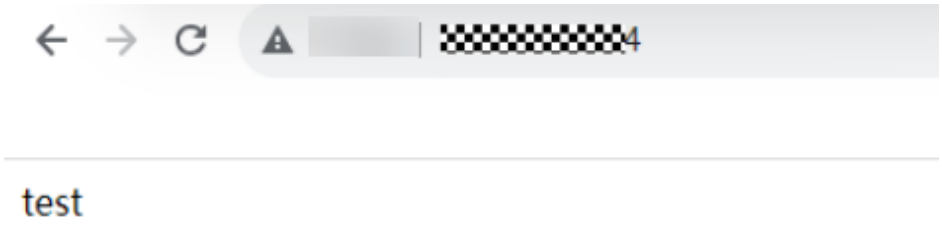
1. Run the following command to install Apache:
   ```
   dnf install httpd httpd-devel
   ```
   After the installation is complete, the Apache configuration file is **/etc/httpd/conf/httpd.conf**.
2. Run the following command to check the Apache version:
   ```
   httpd -v
   ```
   Information similar to the following is displayed:
   ```
   [root@localhost system]# httpd -v
   Server version: Apache/2.4.51 (Unix)
   Server built:   Feb  9 2022 09:00:41
   ```
3. Run the following command to start Apache:
   ```
   systemctl start httpd
   ```
   If **active (running)** is displayed, Apache is started.

> **NOTE**
>
> To enable Apache to automatically start upon system boot, run the following commands in sequence:
>
> ```
> systemctl enable httpd
> systemctl daemon-reload
> ```

Step 2 Verify Apache.

1. Run the following command to create a test page:
   ```
   echo test > /var/www/html/index.html
   ```
2. Enter **http://***<public-IP-address>* in the address box of a browser. If the following information is displayed, the deployment is successful.

   Figure 10-1 Successful deployment

----End

11 Compiling qperf to Support IPv6

Introduction

qperf 0.4.9 in HCE 2.0 does not support IPv6. To use IPv6, you need to obtain qperf 0.4.11 or later from the community.

This section describes how to download and compile the qperf 0.4.11 source code in HCE 2.0.

Preparations

Prepare an ECS and assign a public IP address or EIP to the ECS.

Prerequisites

A yum repository has been configured. For details, see **Configuring Repositories and Installing Software for HCE**.

Procedure

Step 1 Run the following command to download the qperf 0.4.11 source code package:

```
wget https://github.com/linux-rdma/qperf/archive/refs/tags/v0.4.11.tar.gz
```

Step 2 Run the following command to decompress the package:

```
tar -xf qperf-0.4.11.tar.gz
```

After the decompression, the **qperf-0.4.11** directory is generated.

Step 3 Run the following command to install dependencies:

```
dnf install gcc make automake
```

Step 4 Go to the **qperf-0.4.11** directory and run the following commands in sequence for compilation:

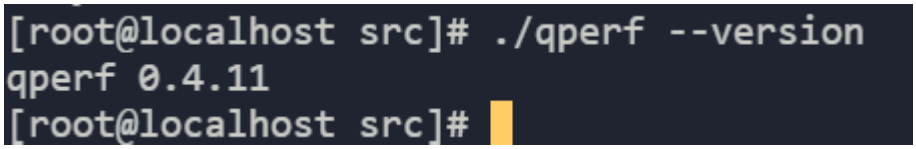
```
./cleanup
./autogen.sh
./configure
make
```

Step 5 Verify the compilation result.

The binary files obtained after compilation are stored in the **qperf-0.4.11/src** directory. Go to the directory and run the **./qperf --version** command. If

information similar to the following is displayed, the execution is successful and qperf is upgraded to 0.4.11.

Figure 11-1 Successful command execution



Step 6 (Optional) Move qperf to the **/usr/bin** directory and verify qperf commands.

Run the following commands in the **qperf-0.4.11/src** directory:

```
cp qperf /usr/bin/
chmod 755 /usr/bin/qperf
qperf --version
```

----End

12 Binding NIC Interrupts to CPU Cores

Introduction

With NIC interrupts bound to cores, data processing requests from each NIC can be allocated to their specified CPU core. This makes network data processing more efficient and improves the data throughput. This section describes how to configure NIC interrupt binding in HCE.

Preparations

Prepare an ECS and assign a public IP address or EIP to the ECS.

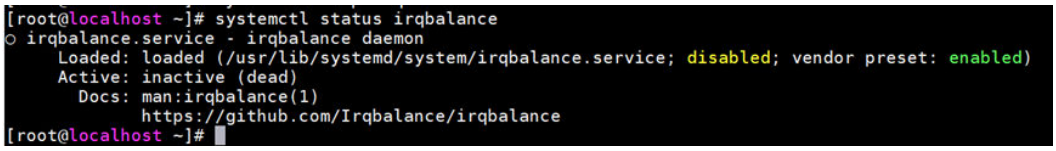
Prerequisites

The irqbalance service is stopped.

Step 1 Check whether irqbalance is stopped.

```
systemctl status irqbalance
```

If **Active: inactive** is displayed, irqbalance is stopped.



Step 2 If **Active: active** is displayed, irqbalance is running. Run the following command to stop it:

```
systemctl stop irqbalance
```

Step 3 (Optional) Prevent irqbalance from starting automatically at system boot.

```
systemctl disable irqbalance
```

----End

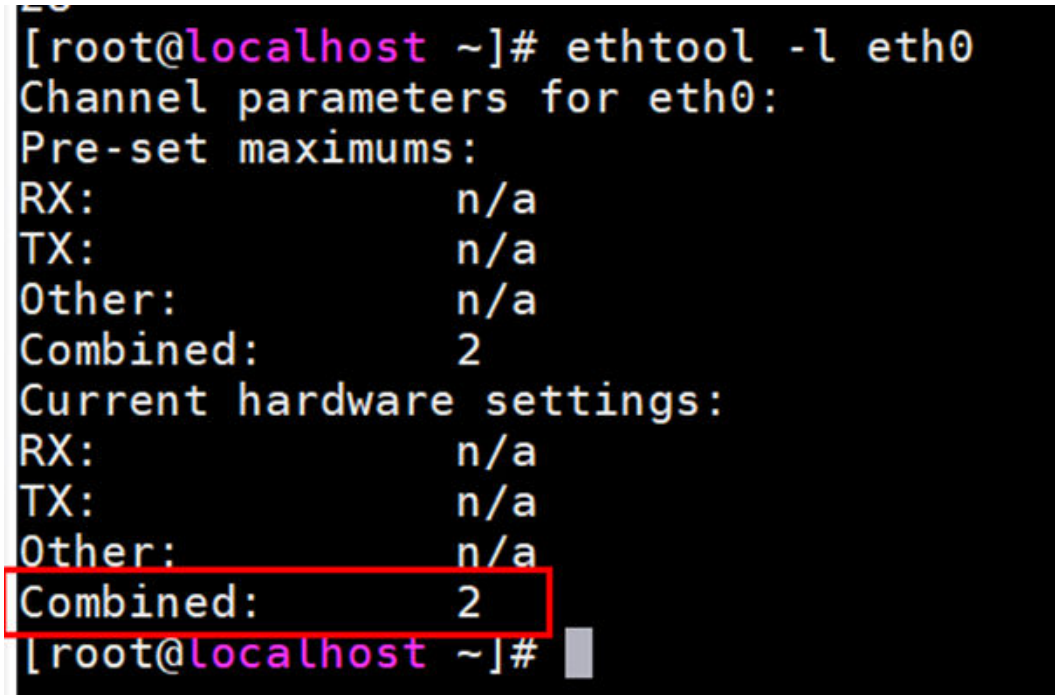
Procedure

Step 1 Run the following command to check the number of NIC queues (in this example, there are two NIC queues. One corresponds to an input interrupt and another corresponds an output interrupt):

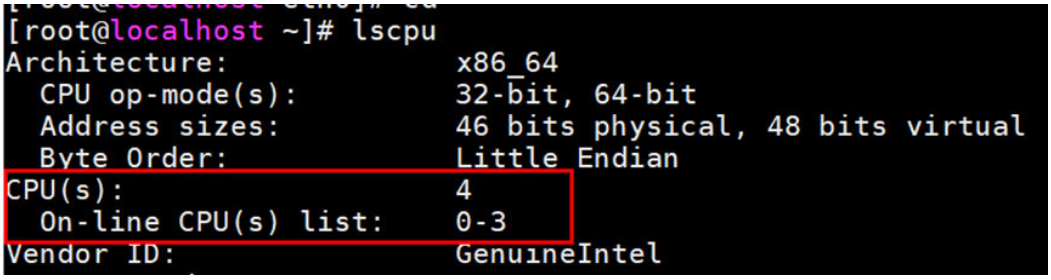

```
ethtool -l eth0
```

NOTE

Replace **eth0** with your actual NIC name. The following steps use **eth0** as an example.



Step 2 Run **lscpu** to check the number of CPU cores. In this example, there are four CPU cores.



Step 3 Check the interrupt corresponding to the current NIC queue.

```
cat /proc/interrupts | grep virtio0 | awk '{print $1 $(NF)}'
```

In this example, data input of eth0 uses interrupts **25** and **27**, and data output uses interrupts **26** and **28**.

NOTE

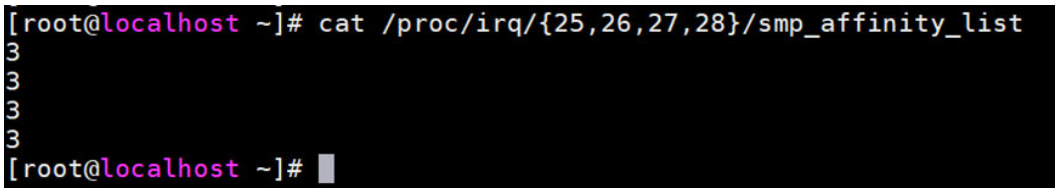
You can run the following command to view the mapping between NICs and VirtIO:

```
ethtool -i eth0 | grep bus-info | awk -F "bus-info:" '{print $2}' | xargs -I {} ls /sys/bus/pci/drivers/virtio-pci/{} | grep virtio
```

Step 4 Check the NIC interrupt binding.

```
cat /proc/irq/{25,26,27,28}/smp_affinity_list
```

In this example, interrupts **25**, **26**, **27**, and **28** are bound to CPU **3**.



Step 5 Manually bind NIC interrupts.

Bind data input interrupts of eth0 to CPU 0 and CPU 1.

```
echo 0 > /proc/irq/25/smp_affinity_list
echo 1 > /proc/irq/27/smp_affinity_list
```

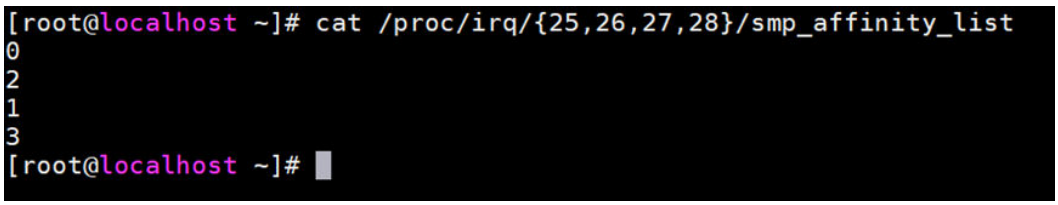
Bind data output interrupts of eth0 to CPU 2 and CPU 3.

```
echo 2 > /proc/irq/26/smp_affinity_list
echo 3 > /proc/irq/28/smp_affinity_list
```

Step 6 Check the binding result.

Run the following command to check the NIC interrupt binding:

```
cat /proc/irq/{25,26,27,28}/smp_affinity_list
```



The NIC interrupts are bound as expected.

----End

