

# System Management



# Foreword

- This chapter focuses on openEuler task, network, and process management, including scheduling.

# Objectives

- Upon completion of this course, you will be familiar with:
  - One-off and scheduled task configuration
  - openEuler network configuration, ensuring hosts can communicate
  - openEuler process viewing and management

# Contents

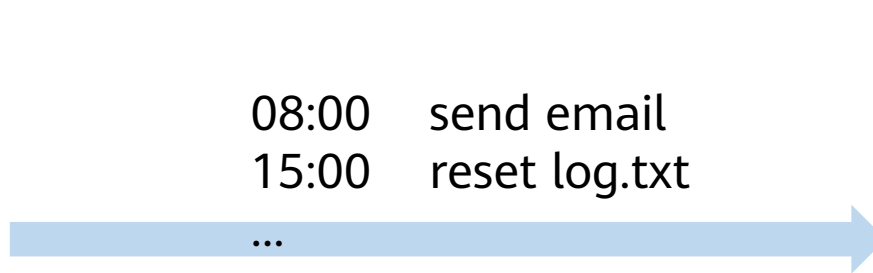
- 1. Task Management**
2. Network Management
3. Process Management

# What Is Task Management?

- During system O&M, a specific task may need to be executed at a specific time.
- For example, sending periodic emails or backing up and clearing log files.
- The content of a task can be regarded as a series of commands, or a script, which needs to be executed at a specific time.



**at, cron, anacron**



**send-email.sh, reset.sh...**

# Task Management Types

- **at:** scheduled task executed once, at a specified time.
- **cron:** task executed periodically for multiple times.

# Using the **at** Command for Scheduled Execution

- The **at** command can be used to specify the time Linux will run scripts.
- The atd process is the daemon process of **at**. It runs in the background when the system is started.
- The atd process periodically checks the **/var/spool/at** directory to get the jobs written by running the **at** command.

```
[root@openEuler ~]# systemctl status atd
```

```
● atd.service - Deferred execution scheduler
```

```
Loaded: loaded (/usr/lib/systemd/system/atd.service; enabled; vendor preset: enabled)
```

```
Active: active (running) since Wed 2020-07-08 11:20:10 CST; 2 weeks 6 days ago
```

```
Docs: man:atd(8)
```

```
Main PID: 2276 (atd)
```

```
Tasks: 1
```

```
Memory: 2.1M
```

```
CGroup: /system.slice/atd.service
```

```
└─2276 /usr/sbin/atd -f
```

```
Jul 08 11:20:10 localhost.localdomain systemd[1]: Started Deferred execution scheduler.
```



# at Command

- The **at** command is used to schedule commands to be executed once only.
- This command requires at least one command and one execution time.
- The **at** command can specify only the time or both time and date.
- **at** command syntax:

```
at [-V] [-q queue] [-f file_name] [-mldbv] time  
at -c job [job...]
```

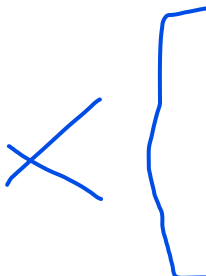


# Setting an Absolute Time

- The scheduled time can be written in any of the following formats:
  - ***hh:mm*** (hour:minute) on the current day. If the time has already passed, the job will be executed the following day.
  - **midnight, noon, teatime** (16:00)
  - 12-hour time followed by **am** or **pm**
  - Time + date (*month day*, ***mm/dd/yy***, or ***dd.mm.yy***)

# Setting a Relative Time

- Relative timing is convenient for commands which need to be executed soon.
  - **now+N minutes, hours, days, or weeks**
  - N indicates the number of minutes, hours, days, or weeks. Another method is to use **today** and **tomorrow** to specify the time to complete commands.
  - Assume that the current time is 12:30 on June 7, 2015, and you want to run a command at 4:30 pm.



at 4:30pm	at now+240 minutes
at 16:30	at 16:30 7.6.15
at 16:30 today	at 16:30 6/7/15
at now+4 hours	at 16:30 Jun 7

- These eight commands all give the same result.

# Execution Permission

- The **at** command can only schedule commands from standard input or from the file specified by the **-f** option.
- If user A switches to user B using the **su** command and then runs **at**, the **at** command's execution result will be sent to user B.
- For other users, running a command or script depends on the **/etc/at.allow** and **/etc/at.deny** files.

```
[root@openEuler ~]# at -f example.sh 15:00
warning: commands will be executed using /bin/sh
job 1 at Wed Jul 29 15:00:00 2020
[root@openEuler ~]# atq
1      Wed Jul 29 15:00:00 2020 a root
```

# Using the cron Command to Run Periodic Commands

- The **at** command can run commands at a scheduled time, but only once.
- If you need to run commands more than once, **cron** is a good helper.



# Cron Running Mechanism

- The cron service searches the **/var/spool/cron** directory for the crontab file named after the username in the **/etc/passwd** file, and then loads the crontab file into the memory.
- It also searches for the **/etc/crontab** (timetable) file, which is written in a specific format.
- If no crontab files are found, the cron service enters sleep mode and releases system resources. The cron service then wakes up every minute to check whether there are commands to be executed.

# Crontab Command

- The **crontab** command is used to **install**, **edit**, **remove**, and **list** crontab files.
- Users can place the sequence of commands to be executed in crontab files. Each user can have their own crontab file.
- Common **crontab** commands:

```
crontab -u //Set a user's cron service. This option is required only when  
the crontab command is run by the root user.
```

```
crontab -l //List details about a user's cron service.
```

```
crontab -r //Remove a user's cron service.
```

```
crontab -e //Edit a user's cron service.
```

# Crontab Files

- A crontab file contains a list of commands meant to be run at certain times.
- Each crontab line has **five** time-and-date fields, followed by a command.
- The fields are separated by **spaces** and **tabs**. The format is as follows:

```
minute hour day-of-month month-of-year day-of-week commands
```



# Crontab File Parameters

minute hour day-of-month month-of-year day-of-week commands

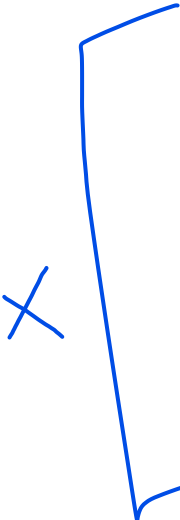
Parameter	Description
minute	Minute (0–59)
hour	Hour (0–23)
day-of-month	Day of a month (1–31)
month-of-year	Month of a year (1–12)
day-of-week	Day of a week (0–6). The value <b>0</b> indicates Sunday.
commands	Commands to be executed

For example, run a series of commands every Monday at 08:00 a.m.

`00 08 * * 1 commands`

# Supplementary Description of Crontab File Parameters

- The parameters on the previous slide cannot be left blank. They must be set.
- An absolute path to the commands shall be provided.
- In addition to numerical values, the following special symbols are allowed:



Parameter	Description
*	All numbers within the value range
/	Step value. For example, */5 indicates every five steps.
-	Value range
,	Several discrete numbers

For example, add **sleepy** to the **/tmp/test.txt** file every two hours from 11:00 p.m. to 8:00 a.m. The corresponding line in the crontab file is as follows:

```
* 23-8/2 * * * echo"sleepy" >> /tmp/test.txt
```

# Editing the Configuration File

- The cron service reads, every minute, **/etc/crontab** and all files in **/var/spool/cron**.
- A crontab file contains user-specific commands. The **/etc/crontab** file contains system-wide commands.
- The file format is as follows:

```
SHELL=/bin/sh
PATH=/usr/bin:/usr/sbin:/sbin:/bin:/usr/lib/news/bin
MAILTO=root          //If an error occurs or data is output, the data is sent to this account by email.
HOME=/
# run-parts01
* * * * root run-parts /etc/cron.hourly // Execute the scripts in /etc/cron.hourly every hour.
02 4 * * * root run-parts /etc/cron.daily // Execute the scripts in /etc/cron.daily once a day.
22 4 * * 0 root run-parts /etc/cron.weekly // Execute the scripts in /etc/cron.weekly once a week.
42 4 1 * * root run-parts /etc/cron.monthly // Execute the scripts in /etc/cron.monthly once a month.
```

# Contents

1. Task Management
- 2. Network Management**
3. Process Management

# Important openEuler Network Concepts (1)

- Device
  - NIC on the host
- Broadcast address
  - Address used to send packets to all hosts on the network segment
- Interface
  - Created by drivers on the devices so that the system can use the devices

# Important openEuler Network Concepts (2)

- Subnet mask
  - Number that distinguishes the network address and the host address within an IP address
- Route
  - Next-hop IP address when IP packets are transmitted across network segments
- Link
  - Device-to-network connection

# Example of openEuler Device Information

```
[root@openEuler ~]# ip addr          // Check devices, including unconfigured network devices.
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp4s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 28:6e:d4:89:70:de brd ff:ff:ff:ff:ff:ff
    inet 192.168.110.246/24 brd 192.168.110.255 scope global dynamic noprefixroute enp4s0
        valid_lft 552233sec preferred_lft 552233sec
    inet6 fe80::fc7e:f7ba:e0d0:2f1f/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default qlen 1000
    link/ether 52:54:00:c7:04:f6 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
        valid_lft forever preferred_lft forever
4: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc fq_codel master virbr0 state DOWN group default qlen 1000
    link/ether 52:54:00:c7:04:f6 brd ff:ff:ff:ff:ff:ff
```



# Configuration File of the openEuler NIC

- Default configuration path of network devices: **/etc/sysconfig/network-scripts/**
- Configuration file: **ifcfg-\***

```
[root@openEuler ~]# ls -l /etc/sysconfig/network-scripts/  
total 4  
-rw-r--r--. 1 root root 308 Jul  8 11:07 ifcfg-enp4s0
```

# NIC Configuration File Parameters

- Parameter values are case insensitive and quotation marks are not required.

Parameter	Description
TYPE	Interface type
BOOTPROTO	Boot-time protocol
ONBOOT	Whether to activate the device at boot-time
IPADDR	IP address
NETMASK	Subnet mask
GATEWAY	Gateway address
BROADCAST	Broadcast address
HWADDR/MACADDR	MAC address. You only need to set one MAC address. MAC addresses cannot conflict with each other.
PEERDNS	Whether to specify the DNS server address. If the DHCP protocol is used, the default value is <b>yes</b> .
DNS{1, 2}	DNS address
USERCTL	User permission control
NAME	Network connection name
DEVICE	Physical interface name

# Modifying the Configuration File

- Use an editor to modify the configuration file. Back up the file before modifying it.
- Modification does not take effect immediately. For the modification to take effect, restart either the NetworkManager service process or the system.

Back up the configuration file.

```
[root@openEuler ~]# cp ifcfg-eth1 ifcfg-eth1.bak
```

Restart the NetworkManager service process.

```
[root@openEuler ~]# systemctl reload NetworkManager
```

# Example of a Minimal NIC Configuration File

- An available NIC configuration file does not need to list all configuration options. This configuration, for example, is valid:

```
TYPE=Ethernet  
BOOTPROTO=static  
NAME=enp0s3  
DEVICE=enp0s3  
ONBOOT=yes  
IPADDR=192.168.56.100  
NETMASK=255.255.255.0
```

# Viewing the IP Address

- Run the **ip** command to view device and address information.

```
[root@openEuler ~]# ip addr show enp4s0
2: enp4s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen
1000
    link/ether 28:6e:d4:89:70:de brd ff:ff:ff:ff:ff:ff
    inet 192.168.110.246/24 brd 192.168.110.255 scope global dynamic noprefixroute enp4s0
        valid_lft 552070sec preferred_lft 552070sec
    inet6 fe80::fc7e:f7ba:e0d0:2f1f/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

No.	Description
1	Status of the current interface. <b>UP</b> : The current interface is enabled. <b>DOWN</b> : The current interface is disabled.
2	Link information: MAC address of the device
3	inet information: IPv4 address and subnet mask
4	Broadcast address, scope, and device name
5	inet6 information: IPv6 information

# NetworkManager

- NetworkManager is a daemon process for dynamically controlling and configuring networks in the system. It is used to keep the current network devices and connections working.
- The nmcli command line tool can be used to control NetworkManager. nmcli is comprehensive, powerful, and sophisticated.
- Key concepts
  - Device: On a network, a device is a network interface.
  - Connection: A connection is a configuration used by a device.
  - A device may have multiple connections, but only one connection at a time can be active.

# Viewing Network Information Using nmcli

- con indicates a connection. The **--active** option can be used to filter active connections.

```
[root@openEuler ~]# nmcli connection show --active
NAME    UUID                                  TYPE    DEVICE
enp4s0  1d859d5a-b0c0-3b49-b153-269e0d4a85ce ethernet enp4s0
virbr0  52137c43-98c9-4f34-8954-d3d1ea89b946 bridge   virbr0
```



# Creating a Connection Using nmcli

1. Create a connection named **Demo** and connect it to the **enp4s0** network port in DHCP mode.
2. Display all connections.
3. Start the created **Demo** connection.
4. Check the status of the connection.

```
[root@openEuler ~]# nmcli connection show --active
NAME    UUID                                  TYPE    DEVICE
enp4s0  1d859d5a-b0c0-3b49-b153-269e0d4a85ce  ethernet enp4s0
virbr0  52137c43-98c9-4f34-8954-d3d1ea89b946  bridge   virbr0
[root@openEuler ~]# nmcli connection show
NAME    UUID                                  TYPE    DEVICE
enp4s0  1d859d5a-b0c0-3b49-b153-269e0d4a85ce  ethernet enp4s0
virbr0  52137c43-98c9-4f34-8954-d3d1ea89b946  bridge   virbr0
[root@openEuler ~]# nmcli connection add con-name 'Demo' type Ethernet ifname enp4s0
Connection 'Demo' (789c723b-cd60-4515-99fc-5074b6f38498) successfully added.
[root@openEuler ~]# nmcli connection show
NAME    UUID                                  TYPE    DEVICE
enp4s0  1d859d5a-b0c0-3b49-b153-269e0d4a85ce  ethernet enp4s0
virbr0  52137c43-98c9-4f34-8954-d3d1ea89b946  bridge   virbr0
Demo    789c723b-cd60-4515-99fc-5074b6f38498  ethernet --
[root@openEuler ~]# nmcli connection up 'Demo'
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/16)
[root@openEuler ~]# nmcli connection show
NAME    UUID                                  TYPE    DEVICE
Demo    789c723b-cd60-4515-99fc-5074b6f38498  ethernet enp4s0
virbr0  52137c43-98c9-4f34-8954-d3d1ea89b946  bridge   virbr0
enp4s0  1d859d5a-b0c0-3b49-b153-269e0d4a85ce  ethernet --
```

# Modifying a Connection Using nmcli (1)

- Run the **nmcli con mod** command to modify a connection. The input parameters are key-value pairs.
- The key is the attribute name, which can be queried by running the **nmcli con show** *[connection\_name]* command. For example:

```
[root@openEuler ~]# nmcli connection show Demo
connection.id:           Demo
connection.uuid:         789c723b-cd60-4515-99fc-5074b6f38498
connection.stable-id:    --
connection.type:         802-3-ethernet
connection.interface-name: enp4s0
connection.autoconnect:  yes
connection.autoconnect-priority: 0
connection.autoconnect-retries: -1 (default)
connection.multi-connect: 0 (default)
connection.auth-retries: -1
connection.timestamp:    1595988280
```

# Modifying a Connection Using nmcli (2)

- The following uses the domain name resolution (DNS) server as an example:

```
[root@openEuler ~]# nmcli connection modify 'Demo' ipv4.dns 192.168.100.250
[root@openEuler ~]# nmcli connection show Demo | grep ipv4.dns
ipv4.dns:                192.168.100.250
ipv4.dns-search:         --
ipv4.dns-options:        ""
ipv4.dns-priority:       0
```

# Routing

- To enable two hosts in different subnets to communicate with each other, a mechanism is required to describe the traffic path between one host and another. This mechanism is called routing. Routing is described using routing entries.
- A routing entry is a pair of predefined addresses: destination and gateway.
- A routing entry allows communication with the destination to be completed through the gateway. Routing entries are listed in a routing table.

# openEuler Route Management and Configuration

- In openEuler, the **route** command is used to view, configure, and manage local routes.
- Routes can also be managed using the **ip** and **nmcli** commands.
- These commands modify the routing table of the system. When the system is started, the routing table is loaded into the memory and maintained by the kernel.

# Using the route Command to View the Routing Table

- Run the **route** command to view the routing table.

```
[root@openEuler ~]# route -n
Kernel IP routing table
Destination    Gateway        Genmask       Flags Metric Ref    Use Iface
0.0.0.0        192.168.110.254 0.0.0.0       UG    100    0      0 enp4s0
192.168.110.0  0.0.0.0        255.255.255.0 U     100    0      0 enp4s0
192.168.122.0  0.0.0.0        255.255.255.0 U      0      0      0 virbr0
```

# Using the route Command to Add a Route

- Add a temporary route to the network segment or host.

```
route [-f] [-p] [Command [Destination] [mask Netmask] [Gateway] [metric Metric]] [if Interface]]
```

- Example:

```
[root@openEuler ~]# route add -net 192.168.101.0 netmask 255.255.255.0 dev enp4s0
```

```
[root@openEuler ~]# route add -host 192.168.100.10 dev enp4s0
```

```
[root@openEuler ~]# route
```

Kernel IP routing table

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
default	_gateway	0.0.0.0	UG	100	0	0	enp4s0
192.168.100.10	0.0.0.0	255.255.255.255	UH	0	0	0	enp4s0
192.168.101.0	0.0.0.0	255.255.255.0	U	0	0	0	enp4s0
192.168.110.0	0.0.0.0	255.255.255.0	U	100	0	0	enp4s0
192.168.122.0	0.0.0.0	255.255.255.0	U	0	0	0	virbr0



# Using the route Command to Delete a Route

- Run the **route del** command to delete a route to a network segment or host.
- Syntax:

```
route del [-net|-host] [netmask Nm] [gw Gw] [[dev] If]
```

- Example:

```
[root@openEuler ~]# route del -host 192.168.100.10 dev enp4s0
```

```
[root@openEuler ~]# route
```

Kernel IP routing table

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
default	_gateway	0.0.0.0	UG	100	0	0	enp4s0
192.168.101.0	0.0.0.0	255.255.255.0	U	0	0	0	enp4s0
192.168.110.0	0.0.0.0	255.255.255.0	U	100	0	0	enp4s0
192.168.122.0	0.0.0.0	255.255.255.0	U	0	0	0	virbr0

# Using the nmcli Command to Configure a Static Route

- Run the **nmcli** command to configure a static route for a network connection.

```
# nmcli connection modify enp3s0 +ipv4.routes "192.168.122.0/24 10.10.10.1"
```

- Run the following interactive commands to configure a static route using the editor:

```
# nmcli con edit type ethernet con-name enp3s0
===| nmcli interactive connection editor |===
Adding a new '802-3-ethernet' connection
Type 'help' or '?' for available commands.
Type 'describe [<setting>.<prop>]' for detailed property description.
You may edit the following settings: connection, 802-3-ethernet (ethernet), 802-1x, ipv4, ipv6, dcb
nmcli> set ipv4.routes 192.168.122.0/24 10.10.10.1
nmcli>
nmcli> save persistent
Saving the connection with 'autoconnect=yes'. That might result in an immediate activation of the
connection.
Do you still want to save? [yes] yes
Connection 'enp3s0' (1464ddb4-102a-4e79-874a-0a42e15cc3c0) successfully saved.
nmcli> quit
```

# Host Name

- A host name uniquely identifies a device in a LAN.
- The device can be a physical or a virtual machine.
- The host name is stored in the **/etc/hostname** file.

```
[root@openEuler ~]# cat /etc/hostname  
openEuler
```

# Setting the Host Name

- Run the **hostname** *new-name* command to temporarily set a host name (valid until next restart).
- Setting a permanent host name: **hostnamectl set-hostname** *new-name*
- Setting a host name by modifying the **hostname** file: write *new-name* to the **/etc/hostname** file.

```
[root@openEuler ~]# hostname
openEuler
[root@openEuler ~]# hostname huawei
[root@openEuler ~]# hostname
huawei
[root@openEuler ~]# hostnamectl set-hostname openEuler01
[root@openEuler ~]# hostname
openEuler01
[root@openEuler ~]# echo "HClA-openEuler" > /etc/hostname
[root@openEuler ~]# hostname
openEuler01
```

# hosts File

- Hosts in a LAN can be accessed through IP addresses.
- When there are a large number of hosts in the LAN, IP addresses are difficult to remember. Therefore, we want to access hosts directly through their host names.
- To do this, we can find the hosts we need in a table that records mappings between host names and IP addresses. This table is **hosts**.

```
[root@openEuler ~]# cat /etc/hosts
127.0.0.1    localhost    localhost.localdomain    localhost4    localhost4.localdomain4
::1         localhost    localhost.localdomain    localhost6    localhost4.localdomain6
```

# Modifying the hosts File

- You can use an editor to modify the **hosts** file in the following format:

```
ip<TAB>domain.com  
192.168.10.20   www.example.com
```

- To delete an entry, add **#** to comment it out. For example:

```
#ip<TAB>domain.com  
#192.168.10.20   www.example.com
```

# Hosts and DNS

- As the number of hosts on a network increases, it becomes difficult for a single **hosts** file to carry a large number of mappings.
- If the IP address mapped to a domain name cannot be found in the **hosts** file, the host submits the domain name to the DNS server, after which the DNS server returns the IP address to the host. This process is called domain name resolution.
- The DNS server is like a public **hosts** file or distributed database.

# Querying a DNS Record (1)

- In openEuler, run the **nslookup** command to query records on the DNS server.
- You can run the **nslookup** command to check whether domain name resolution is normal and diagnose network problems.
- The format of the **nslookup** command is as follows:

```
nslookup domain [dns-server]
```

In the preceding command, domain indicates the domain name to be queried.

[dns-server] specifies the DNS server. This parameter is optional. Common values are 8.8.8.8 and 114.114.114.114.



## Querying a DNS Record (2)

- For example, to query the IP address mapped to **exam.openEuler.com**, run the following command:

```
[root@openEuler ~]# nslookup exam.openEuler.com
```

```
Server:      192.168.100.250
```

```
Address:     192.168.100.250#53
```

```
Name:  exam.openEuler.com
```

```
Address: 192.168.100.250
```

# DNS Resolution Records

- In addition to resolving a domain name to an IP address, the DNS server also supports other types of resolution records.

Record	Description
A	Maps a domain name to an IPv4 address.
CNAME	Maps one domain name to another domain name, to achieve the same domain access using both names.
MX	Sets up an email service that points to the email server address.
NS	Specifies a DNS server to resolve a subdomain name.
TXT	This parameter can be set to any value or left blank. Generally, it is used for verification.
AAAA	Maps a host name (or a domain name) to an IPv6 address.
SRV	Identifies computers hosting specific services.
SOA	Indicates the start of authority. This record includes administrative information about its zone, including the master server among multiple NS records.
PTR	Indicates the reverse of the A record, resolving an IP address to a domain name.
Explicit/Implicit URL forwarding	Maps a domain name to an HTTP/HTTPS address. When a user accesses the domain name, the user is automatically redirected to the target address. The real address is displayed in explicit forwarding and hidden in implicit forwarding.

# Querying Other Records

- Run the **nslookup** command to query various types of records. For example:

```
[root@openEuler ~]# nslookup
> set type=A
> exam.openEuler.com
Server:      192.168.100.250
Address:     192.168.100.250#53

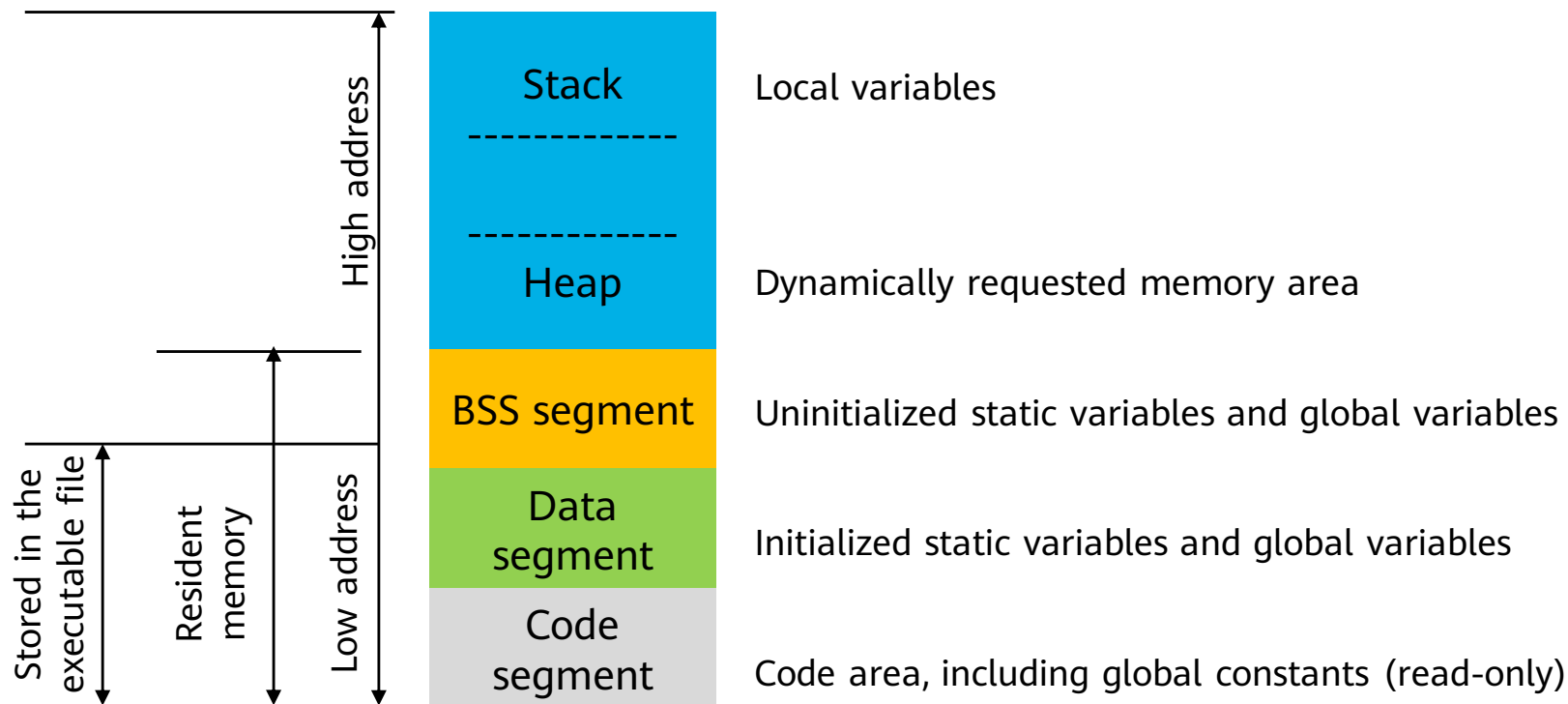
Name: exam.openEuler.com
Address: 192.168.100.250
```

# Contents

1. Task Management
2. Network Management
- 3. Process Management**

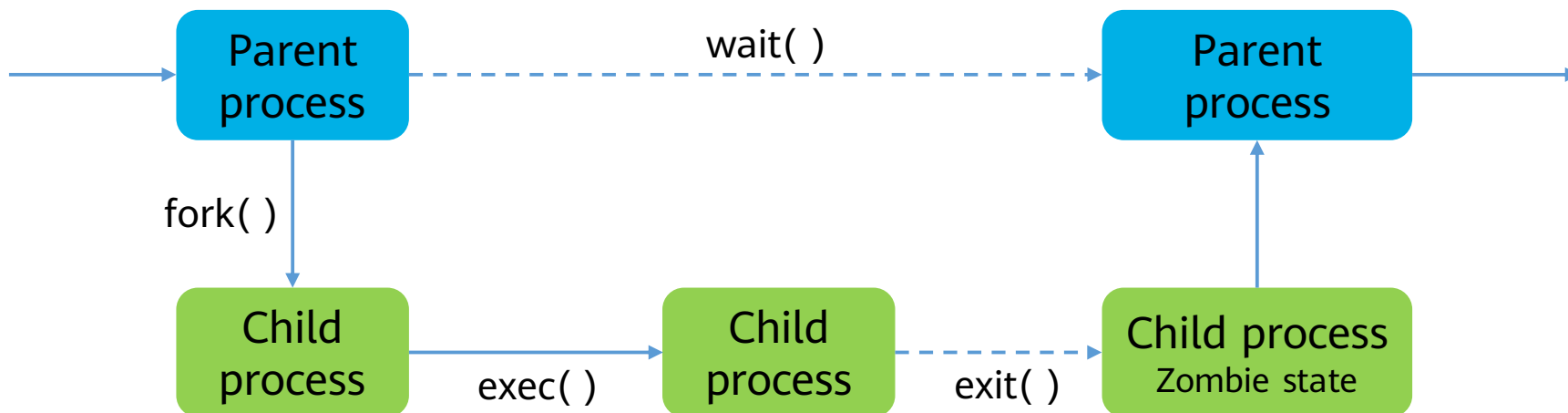
# Introduction to Processes

- A process is an entity of a running program in a computer. It is a specific implementation of that program.
- When a Linux process is created, the system allocates a segment of memory space (a certain logical address space) to the process.

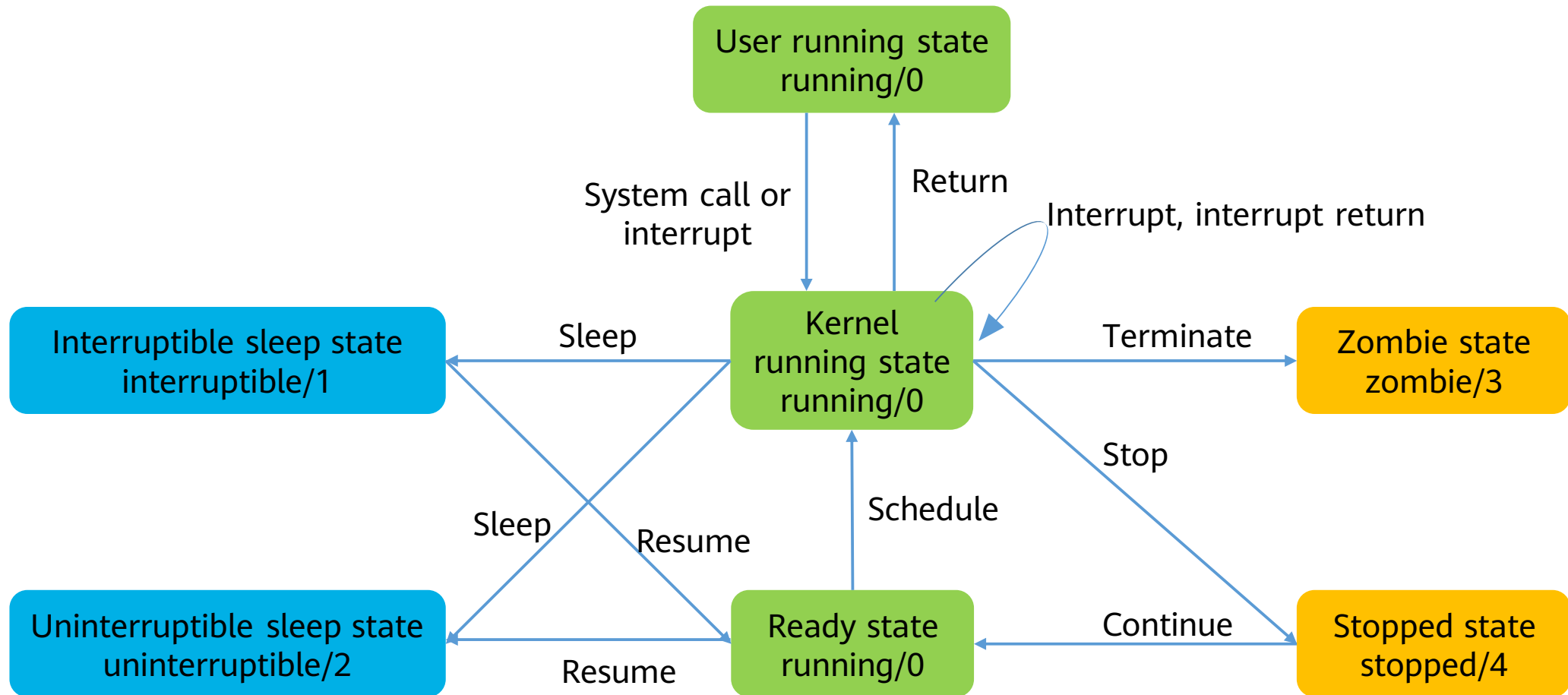


# Process Life Cycle

- Each process has a unique process ID (PID), which is used for tracing the process.
- Any process can create a child process by copying its own address space (fork). The parent process ID (PPID) is recorded in the child process.
- The first system process is systemd, and all the other processes are its descendants.



# Process State



# Process Priority (1)

- A process's CPU resource (time slice) allocation refers to the priority of that process.
- Processes with a higher priority will be executed first.
- Configuring the priority of a process can improve system performance in a multitasking Linux environment.

```
[root@openEuler ~]# ps -l
```

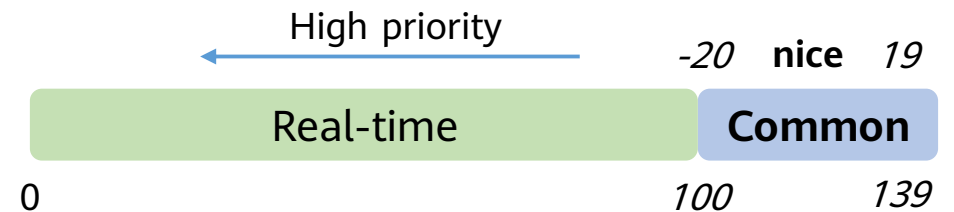
F	S	UID	PID	PPID	C	PRI	NI	ADDR	SZ	WCHAN	TTY	TIME	CMD
0	S	0	686110	686109	0	80	0	-	3383	do_wai	pts/0	00:00:00	bash
0	S	0	686217	686110	0	80	0	-	3386	do_wai	pts/0	00:00:00	bash
0	R	0	688767	686217	0	80	0	-	3399	-	pts/0	00:00:00	ps



## Process Priority (2)

- PRI: process priority, indicating the sequence in which a program will be executed by the CPU. A smaller value indicates higher process priority.
- NI: nice value, indicating a corrected process priority value. It can be understood as "concession".
- The nice value of a process is not the priority of that process. You can, however, adjust the nice value to affect the priority value.

Priority (PRI) Range	Description
0—99	Real-time process
100—139	Non-real-time process



# Adjusting the Priority of a Process

- In openEuler, **nice** and **renice** are used to adjust the nice value of a process, which affects the process priority.

- The syntax of the **nice** command is as follows (the **adjustment** value range is -19 to 20):

```
nice [-n adjustment] [command [arg...]]
```

- To set vi priority to **-18**, for example, run the following command:

```
nice -n -18 vi &
```

- The **adjustment** range of the **renice** command is the same as for **nice**. Objects include the program group **-g**, process **-p**, and user **-u**. The syntax is as follows:

```
renice [-n adjustment] [-] <pid>
```

- To set vi priority to **-12**, for example, run the following command:

```
# renice -n 12 -p 9700  
9700(process ID) old priority -18, new priority 12
```

# Foreground and Background Processes

- Background processes barely interact with users, so they do not need high priority.
  - A daemon process of Linux is a special background process that is independent of the terminal. It periodically executes tasks, or waits to resume executing tasks.
- Foreground processes interact with users, requiring a high response speed and high priority.
  - A foreground process is a process used by a user to control a terminal.

# Controlling Foreground and Background Processes

- The following are common commands used in openEuler to manage processes, including starting, stopping, and switching between foreground and background:
  - **&**: Place an ampersand (&) at the end of a command to execute it in the background.
  - **Ctrl+z**: Moves a running foreground process to the background and suspends it.
  - **Ctrl+c**: Interrupts a running process.
  - **jobs**: Lists processes running in the background.
  - **fg**: Moves a background process to the foreground.
  - **bg**: Resumes a background process.

# Viewing Processes

- When troubleshooting the system, it is important to know the status of processes.
- In openEuler, run the **ps** or **top** command to view the details of different processes.

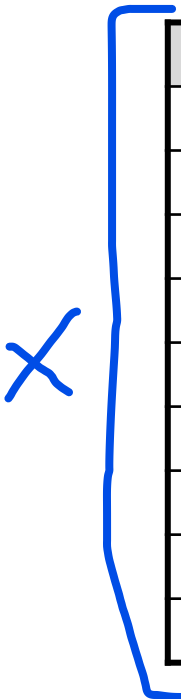
# Using the ps and top Commands to View Processes

- Run the **ps** command to list currently running processes and their accompanying details. The table below describes the fields.
- Run the **ps aux** or **ps -ef** command to view process information.
- The **ps** command lists processes at a specific point in time. The **top** command shows processes in real time.

Field	Description
UID	User identifier
PID	Unique process identifier
%CPU	Process CPU usage
%MEM	Process memory usage
NI	Nice value (process concession)
PRI	Process priority
TTY	Terminal associated with the process
STAT	Process state

# Using Signals to Manage Processes

- In openEuler, processes communicate with each other through signals. The table below lists common signals.
- A process signal is a predefined message that a process recognizes and chooses to act upon or ignore.
- The system administrator needs to know what kind of signal to send to a process, and when.



Signal	Field	Description
1	HUP	Hangup
2	INT	Terminal interrupt
3	QUIT	Terminal quit
9	KILL	Unconditional termination
11	SEGV	Segment error
15	TERM	Termination
17	STOP	Stop executing unconditionally (but do not terminate)
18	TSTP	Terminal stop (but continue to run in the background)
19	CONT	Continue executing, if stopped (STOP or TSTP)

# Managing Processes

- In openEuler, the **kill** and **killall** commands send signals to processes.
- To send signals to a process, the current user must be the owner of that process, or the **root** user.
- **kill** *PID* can send a signal to a process. The TERM signal is sent by default. Use the **-s** option to specify other signals. For example:

```
# kill 3389
```

```
# kill -s HUP 3389
```

signal number can be used instead of field name (eg:1 instead of HUP)

- **killall** *process\_name* can send a signal to a process matching the specified process name. Supporting wildcard characters, **killall** can also send a signal to multiple processes. The following command, for example, will send a signal to all processes starting with 'python'

```
# killall python*
```



# Quiz

1. (Single-answer question) Which of the following commands can dynamically check host resource usage?
  - A. ps
  - B. top
  - C. free
  - D. lscpu
2. (True or false) The **route** command can view or configure host route information. Configured route information will remain valid permanently.

# Summary

This chapter described common management operations in openEuler, including task, network, and process management.

- Task management: how to schedule tasks in the system.
- Network management: basic network knowledge and how to manage network elements in the system.
- Process management: how to view and manage system processes.

# Thank you.

把数字世界带入每个人、每个家庭、  
每个组织，构建万物互联的智能世界。  
Bring digital to every person, home, and  
organization for a fully connected,  
intelligent world.

**Copyright©2022 Huawei Technologies Co., Ltd.  
All Rights Reserved.**

The information in this document may contain predictive statements including, without limitation, statements regarding the future financial and operating results, future product portfolio, new technology, etc. There are a number of factors that could cause actual results and developments to differ materially from those expressed or implied in the predictive statements. Therefore, such information is provided for reference purpose only and constitutes neither an offer nor an acceptance. Huawei may change the information at any time without notice.

