

# System Management and Control Commands


- sudo, on page 2
- setup-center-cli, on page 3
- reboot, on page 6
- date, on page 7
- poweroff, on page 8
- systemctl, on page 9
- crontab, on page 10
- journalctl, on page 11
- unenroll, on page 13

# sudo

Use the **sudo** command to interact with the shell with root privileges, simulating a root login. It provides full administrative access until you exit the shell.

**sudo -i**

For more information, see sudo.

# setup-center-cli

Use the **setup-center-cli** command to configure Cisco Cyber Vision Center.

**setup-center-cli** *COMMAND* `[` *args...* `]`

## Syntax Description

| | |
|---|---|
| **keymap** *KEYMAP* | Configures the keyboard mapping for the Cyber Vision Center.<br>**Usage:** `setup-center-cli keymap KEYMAP`<br>`KEYMAP`: Specifies the keyboard mapping to use (for example, us, fr, de, it, es) |
| **network** | Generates `systemd-networkd` interface files for network configuration.<br>**Usage:** `setup-center-cli network COMMAND [arg...]`<br>**Subcommands:**<br>• `interface`: Generates network interface files.<br>• `dns`: Generates DNS configuration files.<br>• `single-interface`: Sets the Center to single-interface mode (admin interface that is used for webapp and sensor collection).<br>• `firewall`: Controls network allowed list. |
| **pki** *FQDN* | Generates certificates for the Cisco Cyber Vision Center.<br>**Usage:** `setup-center-cli pki FQDN`, where FQDN is the fully qualified domain name of the Center. |
| **renew-certificate** | Renews Center or sensor certificates.<br>**Usage:** `setup-center-cli renew-certificate [--center \| --sensor=<ID>]`<br>**Options:**<br>• `--center`: Renews the Center certificate.<br>• `--sensor=<ID>`: Renews a specific sensor certificate (where `<ID>` is the sensor ID). |
| **webapp-certificate** | Manages the web application's TLS certificate.<br>**Usage:** `setup-center-cli webapp-certificate COMMAND [arg...]`<br>**Subcommands**<br>• `reset`: Resets to the default selfsigned certificate configuration.<br>• `import`: Imports a PKCS#12 certificate file.<br>• `csr`: Uses a Certificate Signing Request (CSR) to configure the web certificate. |

| | |
|---|---|
| **authorized-keys** | Authorizes public SSH keys for access to the Center. If no arguments are provided, it reads keys from standard input.<br>**Usage:** `setup-center-cli authorized-keys [OPTIONS] [FNAME...]`, where FNAME is the filename of the SSH public key to authorize.<br>**Options**<br>• `--user`: Specifies the user for authorized keys (defaults to "cv-admin"). |
| **type** | `type`: Generates configuration files based on the desired Center type.<br>**Usage:** `setup-center-cli type <TYPE>`<br>**Arguments**: `<TYPE>`: Specifies the Center type:<br>• Standalone<br>• Local Center<br>• Global Center<br>**Example:** `setup-center-cli type Local Center.` |
| **ntp** | Generates NTP (Network Time Protocol) configuration files.<br>**Usage:** `setup-center-cli ntp [SERVER...]`, where `[SERVER...]` indicates one or more NTP server configurations.<br>**Example:** `setup-center-cli ntp ntp.example.com,1,secretkey` |
| **center-id** | Changes the Center's unique ID, used for enrollment in a Global Center and for computing component IDs.<br>**Usage:** `setup-center-cli center-id [ID]`, where `[ID]` is the new Center ID. If not provided, the DMI system UUID is used.<br>**Example:** `setup-center-cli center-id new-center-id` |
| **password** | Provides password-related utilities.<br>**Usage:** `setup-center-cli password <COMMAND> [arg...]`<br>**Subcommands**<br>• `check`: Checks password strength.<br>• `setup-user`: Sets the password for the `cv-admin` user.<br>**Example:** `setup-center-cli password check` |
| **import** | Imports configuration settings from a JSON file.<br>**Usage:** `setup-center-cli import <FILE> [-f \| --force]`, where `<FILE>` is the path to the JSON configuration file.<br>**Example:** `setup-center-cli import config.json --force` |

## Command History

| Release | Modification |
|---|---|
| 4.3.0 | This command was introduced. |

This example displays how to check the password strength:

```
root@center100:~# setup-center-cli password check

Password must be at least 16 characters long
```

# reboot

Use the **reboot** command to restart the Cisco Cyber Vision Center.

**reboot**

This example displays how to restart the center:

```
root@center100:~# reboot
Connection to 10.2.3.100 closed by remote host.
```

For more information, see reboot.

# date

Use the **date** command to check the current date on the Cisco Cyber Vision Center CLI or sensor application.

**date**

This example displays how to check the current date on center:

```
root@center100:~# date
Wed Jun  5 11:20:36 UTC 2024
```

This example displays how to check the current date on sensor:

```
sh-5.0# date
Wed Jun  5 11:20:54 UTC 2024
```

For more information, see date.

# poweroff

Use the **poweroff** command to shut down the Cisco Cyber Vision Center server.

**poweroff**

This example displays how to shut down the center:

```
root@center100:~# poweroff
Connection to 10.2.3.100 closed by remote host.
```

For more information, see poweroff.

# systemctl

Use the **systemctl** command to interact and manage the services running on the Cisco Cyber Vision Center server.

**systemctl** { **status** } | { **start** } | { **stop** } | { **restart** } | { **--failed** }

## Syntax Description

| | |
|---|---|
| **--failed** | Lists failed services on the system. |
| **status** <*name of the service*> | Checks the status of the specific service. |
| **restart** <*name of the service*> | Restarts a specific service or all services. |
| **stop** <*name of the service*> | Stops the specific service. |

This example displays how to check failed services:

```
root@center100:~# systemctl --failed
0 loaded units listed.
```

This example displays how to check the status of the "sbs-backend.service"service:

```
root@center100:~# systemctl status sbs-backend.service
sbs-backend.service - Cisco Cyber Vision Center Backend
     Loaded: loaded (/lib/systemd/system/sbs-backend.service; enabled; vendor preset:
enabled)
     Active: active (running) since Wed 2024-06-05 16:32:32 UTC; 2s ago
   Main PID: 5617 (sbs-backend-sta)
      Tasks: 22 (limit: 77128)
     Memory: 92.4M
```

This example displays how to restart all `sbs` services:

```
root@center100:~# systemctl restart sbs-services.target
```

This example displays how to restart the "sbs-backend.service" service:

```
root@center100:~# systemctl restart sbs-backend.service
```

For more information, see systemctl.

# crontab

**Crontab** is a configuration file that schedules commands or scripts to run automatically at specific intervals.

```
Usage:
 crontab [options] file
 crontab [options]
 crontab -n [hostname]

Options:
 -u <user>  define user
 -e         edit user's crontab
 -l         list user's crontab
 -r         delete user's crontab
 -i         prompt before deleting
 -n <host>  set host in cluster to run users' crontabs
 -c         get host in cluster to run users' crontabs
 -V         print version and exit
 -x <mask>  enable debugging
```

Syntax and Descriptions

Each line in a crontab file follows a specific syntax:

1. **Cron Expression:** The line begins with a cron expression consisting of five fields:
   - Minute (0-59)
   - Hour (0-23)
   - Day of the Month (1-31)
   - Month (1-12 or Jan-Dec)
   - Day of the Week (0-6 or Sun-Sat)

   These fields represent the time and date when the scheduled command should be executed.
2. **Command:** The cron expression is followed by the command or script to be executed.



**Note** If both the "day of month" and "day of week" fields are restricted (i.e., not "*"), then either or both of these fields must match the current day for the job to be executed.

This is a crontab configuration for purging the components that are inactive for 90 days:

```
5 * * * * sbs-db purge-components --inactive-days 90
```

This is a crontab configuration for deleting the table content every two days at midnight.

```
0 0 */2 * * sudo sbs-db-toolbox exec 'TRUNCATE TABLE dns_request;'
```

For more information, see crontab.

# journalctl

Use the **journalctl** command to interact and search through the log entries that are stored in the journal.

**journalctl** [ **-r** ] [ **--since** ] [ **-f** ] [ **-p err** ] [ **-u** <*servicename*> ]

## Syntax Description

| | |
|---|---|
| **-r** | Displays the latest logs first. |
| **--since** | Displays logs within a specified time range. |
| **-f** | Displays live logs for live troubleshooting. |
| **-p err** | Fecthes only the error logs. |
| **-u** <*servicename*> | Displays the logs for a specific service. |
| **--boot=0** | Displays the logs from the last system boot. |

This example displays how to extract Linux journal for the "sbs-burrow" service:

```
root@center100:~# journalctl -u sbs-burrow
-- Logs begin at Mon 2024-05-13 12:28:06 UTC, end at Thu 2024-06-06 12:51:44 UTC. --
May 14 03:14:31 center burrow[6748]: burrow flow table analyzed in 0.00 secs
[caller=flowtable_analyzer.go:153]
May 14 03:14:32 center burrow[6748]: burrow -- 1 files handled in 0.020166 seconds
[caller=interfacer.go:71]
May 14 03:14:37 center burrow[6748]: burrow flow table analyzed in 0.01 secs
[caller=flowtable_analyzer.go:153]
May 14 03:14:37 center burrow[6748]: burrow flow table analyzed in 0.00 secs
[caller=flowtable_analyzer.go:153]
May 14 03:14:37 center burrow[6748]: burrow flow table analyzed in 0.00 secs
[caller=flowtable_analyzer.go:153]
```

This example displays how to extract the live logs for the "sbs-burrow" service:

```
root@center100:~# journalctl -fu sbs-burrow
-- Logs begin at Mon 2024-05-13 12:28:06 UTC. --
Jun 06 12:52:31 center burrow[147743]: burrow flow table analyzed in 0.00 secs
[caller=flowtable_analyzer.go:153]
Jun 06 12:52:31 center burrow[147743]: burrow flow table analyzed in 0.01 secs
[caller=flowtable_analyzer.go:153]
Jun 06 12:52:31 center burrow[147743]: burrow -- 3 files handled in 0.049746 seconds
[caller=interfacer.go:71]
Jun 06 12:52:36 center burrow[147743]: burrow flow table analyzed in 0.00 secs
[caller=flowtable_analyzer.go:153]
Jun 06 12:52:36 center burrow[147743]: burrow -- 1 files handled in 0.013072 seconds
[caller=interfacer.go:71]
Jun 06 12:52:46 center burrow[147743]: burrow flow table analyzed in 0.00 secs
[caller=flowtable_analyzer.go:153]
Jun 06 12:52:46 center burrow[147743]: burrow -- 1 files handled in 0.009560 seconds
[caller=interfacer.go:71]
Jun 06 12:52:51 center burrow[147743]: burrow flow table analyzed in 0.00 secs
[caller=flowtable_analyzer.go:153]
Jun 06 12:52:51 center burrow[147743]: burrow flow table analyzed in 0.00 secs
[caller=flowtable_analyzer.go:153]
Jun 06 12:52:51 center burrow[147743]: burrow -- 2 files handled in 0.028321 seconds
[caller=interfacer.go:71]
```

This example displays how to extract the logs for the "sbs-burrow" service:

```
root@center100:~# journalctl -u sbs-burrow-- Logs begin at Mon 2024-05-13 12:28:06 UTC, end
 at Thu 2024-06-06 12:51:44 UTC. --
May 14 03:14:31 center burrow[6748]: burrow flow table analyzed in 0.00 secs
[caller=flowtable_analyzer.go:153]
May 14 03:14:32 center burrow[6748]: burrow -- 1 files handled in 0.020166 seconds
[caller=interfacer.go:71]
May 14 03:14:37 center burrow[6748]: burrow flow table analyzed in 0.01 secs
[caller=flowtable_analyzer.go:153]
May 14 03:14:37 center burrow[6748]: burrow flow table analyzed in 0.00 secs
[caller=flowtable_analyzer.go:153]
May 14 03:14:37 center burrow[6748]: burrow flow table analyzed in 0.00 secs
[caller=flowtable_analyzer.go:153]
```

For more information, see journalctl.

# unenroll

Use the **unenroll** command to unenroll a local Center from the Cisco Cyber Vision Global Center synchronization. It effectively removes the local Center's data from the global Center and prepares the local Center for future enrollment or replacement.

**/opt/sbs/bin/unenroll**

## Command History

| Release | Modification |
|---|---|
| 5.2.0 | This command was introduced. |

This example displays how to unenroll a local Center:

```
root@center100:~# opt/sbs/bin/unenroll
08/01/2026 18:38:07 +0000 unenroll INFO Center type: standalone
                                                      caller=postgres.go:527
08/01/2026 18:38:07 +0000 unenroll INFO Non-enrolled center, exiting
                                                            caller=main.go:56
root@Center:~#
```