Red Hat Enterprise Linux 10

Interacting with the command-line assistant powered by RHEL Lightspeed

Leverage the AI-driven expertise of the command-line assistant powered by RHEL Lightspeed to help you configure, manage, and troubleshoot RHEL

Last Updated: 2026-02-03

Red Hat Enterprise Linux 10 Interacting with the command-line assistant powered by RHEL Lightspeed

Leverage the AI-driven expertise of the command-line assistant powered by RHEL Lightspeed to help you configure, manage, and troubleshoot RHEL

Legal Notice

Abstract

With the command-line assistant powered by RHEL Lightspeed, you can get expert advice and assistance with managing RHEL right from your command line, all by using natural language. You can also use the RHEL Model Context Protocol (MCP) server for a more protected and context-aware way for AI models to perform system administration and troubleshooting on RHEL systems, such as log and performance analysis. The generative AI that powers the assistant incorporates information from the RHEL product documentation and Red Hat Knowledgebase, and can help you understand, configure, and troubleshoot your RHEL systems.

Table of Contents

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your feedback on our documentation. Let us know how we can improve it.

Submitting feedback through Jira (account required)

1. Log in to the Jira website.
2. Click **Create** in the top navigation bar.
3. Enter a descriptive title in the **Summary** field.
4. Enter your suggestion for improvement in the **Description** field. Include links to the relevant parts of the documentation.
5. Click **Create** at the bottom of the dialogue.

CHAPTER 1. INTRODUCING RED HAT LIGHTSPEED FOR RHEL SYSTEMS

The Red Hat Lightspeed intelligent functionalities can help you to manage your system environment in a more accessible way, whether you are less experienced with RHEL or already have experience.

1.1. THE COMMAND-LINE ASSISTANT POWERED BY RHEL LIGHTSPEED

The command-line assistant powered by RHEL Lightspeed is an optional AI tool available within the RHEL command-line interface, and includes information from the Red Hat Knowledge Centered Service (KCS) articles, RHEL documentation, and other Red Hat resources.

You can use the assistant to get help, for example, with the following activities:

- Answering RHEL-related questions
- Assistance with troubleshooting and fixing issues
- Understanding log files
- Asking for recommendations

You can use the command-line assistant powered by RHEL Lightspeed for interactive workflows to solve issues, implement new RHEL features, find information, and more. For example, you can run a command and then use the command-line assistant to help you understand the output and possible next steps. Or, you can ask a question on an SSH error, receive suggestions, and ask another question to continue diagnosing the problem.

You can interact with the command-line assistant powered by RHEL Lightspeed by using plain language instead of using complex commands as you might when using a standard command-line interface.

The command-line assistant powered by RHEL Lightspeed does not require direct internet connectivity. This is helpful if you do not want to have every RHEL system directly connected to a service over the internet. Instead, you can proxy all of the requests from RHEL systems through a single proxy system that is connected to the internet.

The RHEL Lightspeed command-line assistant follows the RHEL lifecycle. For information on supported versions and the associated policies, see the Red Hat Enterprise Linux Life Cycle for information on supported versions and the associated policies.

IMPORTANT

The command-line assistant does not have direct access to the information about the system it is running on. However, you can include information about your environment in the messages you input and which are then sent to the LLM provider. For example, the assistant is unable to provide answers on the free memory available on the system that it runs. Instead, the command-line assistant responds with information about a command that you can run to determine how much free memory there is.

1.2. HOW THE COMMAND-LINE ASSISTANT PROCESSES YOUR DATA

While using the command-line assistant interface, you input messages that the command-line assistant transforms and sends to the LLM provider you have configured for your environment. These messages can contain information about aspects of your environment.

Do not enter information into the command-line assistant interface that you do not want to send to the LLM provider.

By using the command-line assistant powered by RHEL Lightspeed, you agree that Red Hat may use all of the messages that you exchange with the LLM provider. The feature is not intended to process personal information, and by using the command-line assistant powered by RHEL Lightspeed you agree to not include any personal information while using the command-line assistant. Support for AI features is provided only for the components that are provided by Red Hat.

CHAPTER 2. INSTALLING THE COMMAND-LINE ASSISTANT POWERED BY RHEL LIGHTSPEED

Before you can start using the command-line assistant powered by RHEL Lightspeed, you must install it through the official RHEL repositories.

The command-line assistant is supported in the following architectures:

- AMD and Intel 64-bit (**x86_64**)
- ARM64 (**aarch64**)
- IBM Z (**s390x**)
- IBM POWER systems (**ppc64**)

To access the command-line assistant powered by RHEL Lightspeed, install it by using RHEL repositories. Do not use **pip install command-line-assistant** because Red Hat does not support this installation option.

Prerequisites

- You have a subscribed RHEL system. For more information, see Getting Started with RHEL System Registration documentation.

Procedure

- On your RHEL system, run the following command:

  ```
  $ sudo dnf install command-line-assistant
  ```

Verification

- Verify that the installation works by running the command-line assistant. For example:

  ```
  $ c "How to install python?"
  ```

  The output looks similar to the following example:

  ```
  +*+ Asking Red Hat Lightspeed

  To install python....
  ```

- Disable the color output in the command-line assistant. For example:

  ```
  $ NO_COLOR=1 c "How to install python?"
  ```

CHAPTER 3. PROVISIONING THE COMMAND-LINE ASSISTANT TO RHEL DEPLOYMENTS WITH RED HAT SATELLITE

You can install the command-line assistant powered by RHEL Lightspeed on your host registered to Red Hat Satellite. For that, update the command-line assistant endpoint so that the RHEL system registered to Satellite can proxy the command-line assistant by using the Satellite Server.

Prerequisites

- Your system is registered with Red Hat Lightspeed for Red Hat Enterprise Linux.
- You are using the command-line assistant on a host registered to a Satellite 6.17+ server or later versions.
- The Satellite Server must be connected to the internet.
- You have enabled the AppStream repository on the host to be able to install the command-line assistant.

Procedure

1. Install command-line assistant powered by RHEL Lightspeed on your registered host.

   ```
   $ sudo dnf install command-line-assistant
   ```

2. Locate and open the **/etc/xdg/command-line-assistant/config.toml** file.
3. In the **config.toml** file, replace the endpoint configuration option to point to your Satellite or Capsule hostname, for example:

   ```
   #The endpoint points to an API server.
   endpoint = "https://satellite.example.server.com/api/lightspeed/v1"
   ```

4. Save the changes in the **config.toml** file.
5. Restart the command-line assistant daemon (**clad**) for the changes to take effect:

   ```
   $ sudo systemctl restart clad
   ```

Troubleshooting

If the command-line assistant cannot trust the Red Hat Satellite certificate authority (CA), it cannot establish a secure connection to the Satellite Server. As a result, the assistant does not function. To work around this problem:

1. Copy the Satellite CA certificate to the system truststore:

   ```
   $ sudo cp /etc/rhsm/ca/katello* /etc/pki/ca-trust/source/anchors/
   ```

2. Update the CA trust database:

   ```
   $ sudo update-ca-trust
   ```

CHAPTER 4. USING THE COMMAND-LINE ASSISTANT COMPONENT FUNCTIONS IN IMAGE MODE FOR RHEL

You can use the command-line assistant component functions in Image Mode for RHEL during container builds and at runtime to perform necessary system tasks while maintaining the security and reliability benefits of read-only file systems on **x86_64** and **aarch64** architectures.

Prerequisites

- The **container-tools** meta-package is installed.
- You have a subscribed RHEL system. For more information, see the Getting Started with RHEL System Registration documentation.

Procedure

1. Create a **Containerfile**:

```
$ cat Containerfile
FROM registry.redhat.io/rhel10/rhel-bootc:latest
RUN <<EORUN
set -euxo pipefail
dnf install -y command-line-assistant
dnf clean all
rm -rf /var/cache/dnf/
bootc container lint
EORUN
```

2. Build the ***<image>*** image by using **Containerfile** in the current directory:

```
$ podman build -t quay.io/<namespace>/<image>:<tag> .
```

3. Build a disk image from the container image. The following example creates a QCOW2 image:

```
$ sudo podman run \
   --rm \
   -it \
   --privileged \
   --pull=newer \
   --security-opt label=type:unconfined_t \
   -v ./config.toml:/config.toml:ro \
   -v ./output:/output \
   -v /var/lib/containers/storage:/var/lib/containers/storage \
   registry.redhat.io/rhel10/bootc-image-builder:latest \
   --type qcow2 \
   --config /config.toml \
   quay.io/<namespace>/<image>:<tag>
```

- ***<namespace>***: your container registry namespace.
- ***<image>***: your image name.
- ***<tag>***: your image tag, for example, latest.

4. Boot a machine with the QCOW2 disk image. For instructions, see Deploying a container image by using KVM with a QCOW2 disk image.

Verification

- Verify that the installation works by running the command-line assistant. For example:

  ```
  $ c "How to deploy a QCOW2 disk image?"
  ```

  The output looks similar to the following example:

  ```
  +*+ Asking RHEL Lightspeed

  To deploy a QCOW2 disk image....
  ```

CHAPTER 5. USING THE COMMAND-LINE ASSISTANT POWERED BY RHEL LIGHTSPEED ON RHEL SYSTEMS

The command-line assistant powered by RHEL Lightspeed can help you to interact with and find information about RHEL by using the command-line interface. Use the command-line assistant to answer RHEL-related questions, troubleshoot, interpret log entries, and perform many other tasks.

WARNING

Do not rely on the results from AI tools without human review. Always check the AI and LLM-generated responses for accuracy before using the generated suggestions.

5.1. CONSULTING THE COMMAND-LINE ASSISTANT

To use the command-line assistant powered by RHEL Lightspeed, use the **c** command followed by your questions in quotation marks.

WARNING

Do not enter the following types of data when using the command-line assistant, because the assistant is not intended to process data such as:

- Personal information
- Business-sensitive information
- Confidential information
- System data information

Prerequisites

- You have installed the command-line assistant powered by RHEL Lightspeed. See the Installing the command-line assistant powered by RHEL Lightspeed documentation.

Procedure

- Use the **c** command, followed by your questions in quotation marks. The following are examples of prompts that you can use to interact with the command-line assistant when using your RHEL system:
  - Ask a data-based question:

    ```
    $ c "What is RHEL"
    ```

-

- Request information on how to troubleshoot a problem with SSHD:

  ```
  $ c "how to troubleshoot sshd failing to start"
  ```

- Request information on how to find files under **/etc**:

  ```
  $ c "how do I find all the files in the /etc/ that have been modified in the last hour"
  ```

5.2. ATTACHING A FILE TO YOUR QUESTIONS FOR THE COMMAND-LINE ASSISTANT

You can attach a file to the command-line assistant powered by RHEL Lightspeed. In response, the assistant can provide a tailored answer based on that file.

Prerequisites

- You have installed the command-line assistant powered by RHEL Lightspeed. See the Installing the command-line assistant powered by RHEL Lightspeed documentation.
- You have created a readable file with the information that you want the command-line assistant to use. The following examples use the **storage_info** file, which contains information about the volume group, logical volumes and file systems on another system.

Procedure

- To replicate the storage configuration on another system, first save the layout of the volume group, logical volumes, and file systems to a <storage_info> file. Then, use the command-line assistant to read this file and to incorporate that information into the next steps.

  ```
  $ c --attachment <storage_info>
  ```

- Optionally, use the short version of the **attachment** command:

  ```
  $ c -a <storage_info>
  ```

- Combine the attachment with a question:

  ```
  $ c --attachment <storage_info> "how can I replicate the storage configuration in another system"
  ```

5.3. CHECKING YOUR HISTORY OF INTERACTIONS WITH THE COMMAND-LINE ASSISTANT

To view answers that the command-line assistant powered by RHEL Lightspeed provided you with previously, you can access your conversation history with the assistant.

Prerequisites

- You have installed the command-line assistant powered by RHEL Lightspeed. See the Installing the command-line assistant powered by RHEL Lightspeed documentation.

Procedure

- Use the **c history** command to parse your conversation history. For example:
  - Fetch all user history:

    ```
    $ c history --all
    ```

  - Access the first conversation from history:

    ```
    $ c history --first
    ```

  - Access the last conversation from history:

    ```
    $ c history --last
    ```

  - Filter your history conversation to search for a term to retrieve all questions and answers related to that word:

    ```
    $ c history --filter "podman"
    ```

  - Clear all the user history:

    ```
    $ c history --clear
    ```

5.4. REDIRECTING A COMMAND OUTPUT TO THE COMMAND-LINE ASSISTANT

To use the command-line assistant powered by RHEL Lightspeed to understand the output of a command, you can redirect the output of the command to the command-line assistant.

Prerequisites

- You have installed the command-line assistant powered by RHEL Lightspeed.

Procedure

- Redirect the log file output containing information that you want to understand to the command-line assistant:

  ```
  $ cat <log_file.log> | c
  ```

- If the error or log that you provided to the command-line assistant do not provide enough information, combine the redirect output with a question:

```
$ cat <log_file_error.log> | c "how do I solve this?"
```

You can also redirect a question:

```
$ echo "how do I solve this?" | c -a <log_file_error.log>"
```

5.5. ENABLING THE COMMAND-LINE ASSISTANT TO CAPTURE YOUR TERMINAL ACTIVITY

You can use the command-line assistant powered by RHEL Lightspeed to reference commands that you previously ran.

WARNING

If you add terminal context in a request and there are no previously captured commands, the command will fail. You can only add context from the terminal while the capture mode is enabled.

Prerequisites

- You have installed the command-line assistant powered by RHEL Lightspeed. See the Installing the command-line assistant powered by RHEL Lightspeed documentation.

Procedure

1. Enable the terminal capture for your current terminal session:

```
$ c shell --enable-capture
```

2. Run at least one command before you reference previous commands.

3. Reference the output of a previously run command. For example, to reference the previous command, run:

```
$ c -w 1 "what_is_this"
```

   - To reference the second to previous command, run:

```
$ c -w 2 "what_is_this"
```

4. To stop terminal capture, press the **Ctrl + D** keys on your keyboard.:

5.6. SUBMITTING FEEDBACK ABOUT THE COMMAND-LINE ASSISTANT RESPONSES

Help improve the quality of the responses that you receive when you interact with the command-line assistant powered by RHEL Lightspeed by submitting feedback on its responses.

Prerequisites

- You have installed the command-line assistant powered by RHEL Lightspeed.

Procedure

- Run the following command:

  ```
  $ c feedback
  ```

CHAPTER 6. USING A CONTAINERIZED RHEL COMMAND-LINE ASSISTANT FOR DISCONNECTED ENVIRONMENTS

RHEL command-line assistant for disconnected environments is available as a Developer Preview. You can use it in air-gapped networks, remote locations, and other environments with limited internet connectivity.

This version is packaged in a UBI-based container and includes simplified configuration, orchestration, and basic lifecycle management.

WARNING

Changing the RHEL Offline Container model to unsupported models can allow the execution of arbitrary code or compromise the integrity of your Red Hat Enterprise Linux (RHEL) systems.

IMPORTANT

Red Hat Enterprise Linux (RHEL) command-line assistant is Developer Preview software only. Developer Preview software is not supported by Red Hat in any way and is not functionally complete or production-ready. Do not use Developer Preview software for production or business-critical workloads. Developer Preview software provides early access to upcoming product software in advance of its possible inclusion in a Red Hat product offering. Customers can use this software to test functionality and provide feedback during the development process. This software might not have any documentation, is subject to change or removal at any time, and has received limited testing. Red Hat might provide ways to submit feedback on Developer Preview software without an associated SLA.

For more information about the support scope of Red Hat Developer Preview software, see Developer Preview Support Scope.

With the Red Hat Enterprise Linux (RHEL) command-line assistant for disconnected environments, you can create a container image targeted for a single system. This container image includes the following components:

- An **installer** container to pull the other required containers, install the **rhel-cla** command, and optionally create a **systemd** service.
- An **rlsapi** container to provide the endpoint used by the command-line assistant client.
- A **rag-database** container that contains the retrieval-augmented generation (RAG) database, which supplements the knowledge of the LLM with additional data, such as the RHEL documentation.
- A **ramalama** container that runs the LLM inference service.
- The command-line assistant powered by Red Hat Enterprise Linux Lightspeed.
  The container image includes the **Phi-4-mini-instruct-Q4_K_M** model, and serves as the default model for the **ramalama** container

The container image is intended for local, disconnected use on a single system. It is not designed for multi-system deployments or large-scale environments.

6.1. INSTALLING THE RHEL COMMAND-LINE ASSISTANT IN DISCONNECTED ENVIRONMENTS

Install the RHEL command-line assistant in a disconnected environment by using the installation container images available on the Red Hat Container Registry and Quay.io. Manually transfer the **installer** image and its dependency images to the local storage of the offline system in the disconnected setup environment.

Prerequisites

- Your RHEL system is registered to a Red Hat Satellite subscription. For additional details, see the Does the offline RHEL command-line-assistant require the system to be registered? article.
- You have Podman installed.
- You have the **container-tools** meta-package installed.
- You have created the **$HOME/.config** and **$HOME/.local/bin** folders.

Procedure

1. On the connected RHEL system, complete the following steps:
   a. Log in to the Red Hat registry:
      ```
      $ podman login registry.redhat.io
      ```
   b. Download the **installer** and all dependency images:
      ```
      $ podman pull registry.redhat.io/rhel-cla/installer-rhel10:latest
      $ podman pull registry.redhat.io/rhel-cla/rag-database-rhel10:latest
      $ podman pull quay.io/ramalama/ramalama:latest
      $ podman pull registry.redhat.io/rhel-cla/rlsapi-rhel10:latest
      ```
   c. Save all the images as a single **tar** file:
      ```
      $ podman save -o rhel-cla-installer-images.tar \
         registry.redhat.io/rhel-cla/installer-rhel10:latest \
         registry.redhat.io/rhel-cla/rag-database-rhel10:latest \
         quay.io/ramalama/ramalama:latest \
         registry.redhat.io/rhel-cla/rlsapi-rhel10:latest
      ```
2. Transfer the **rhel-cla-installer-images.tar** file to the disconnected system by using a USB drive, network share, or other transfer method.
3. On the disconnected system, load the image:
   ```
   $ podman load -i rhel-cla-installer-images.tar
   ```

4. On the disconnected system, run the **installer** container. This command uses the loaded image and installs the offline RHEL command-line assistant on your system.

   ```
   $ podman run -u : --rm \
   -v $HOME/.config:/config:Z \
   -v $HOME/.local/bin:/config/.local/bin:Z \
   rhel-cla/installer-rhel10:latest
   ```

5. Optional: To automatically start the offline RHEL command-line assistant containers each time the system boots, add the **install-systemd** argument to the **podman run** command:

   ```
   $ podman run -u : --rm \
   -v $HOME/.config:/config:Z \
   -v $HOME/.local/bin:/config/.local/bin:Z \
   registry.redhat.io/rhel-cla/installer-rhel10:latest install-systemd
   ```

6. Install the RHEL **command-line-assistant client** package on your offline RHEL system:

   ```
   # sudo dnf install command-line-assistant
   ```

7. Update the **endpoint** configuration option.

   a. Open the **/etc/xdg/command-line-assistant/config.toml** file in a text editor.

   b. Locate the **endpoint** variable and change its value to the address of the system hosting your offline RHEL command-line assistant containers. For example, to connect to the local system:

      ```
      endpoint = "http://127.0.0.1:8000/v1"
      ```

8. Restart the **clad** service to apply the new configuration:

   ```
   $ sudo systemctl restart clad
   ```

   After the installation is complete, you can use the **rhel-cla** command to start, stop, uninstall, and check the status of the offline RHEL command-line assistant.

Verification

1. Check if the offline RHEL command-line assistant is running:

   ```
   $ rhel-cla status
   ```

2. Verify that the container is running:

   ```
   $ podman ps
   CONTAINER ID   IMAGE         COMMAND     CREATED          STATUS
   ba655e5efdcd   installer-rh… /sbin/init  30 seconds ago   Up 29 seconds
   ```

   It might take about a minute for the **rlsapi** and the **ragdb** containers to fully start.

3. If the assistant is not running, start it:

```
$ rhel-cla start
```

4. Check if the client is correctly configured by asking the following question:

```
$ c "What is an immutable file?"

+ Asking Red Hat Lightspeed
This feature uses AI technology. Do not include any personal information or other sensitive
information in your input. Interactions may be used to improve Red Hat's products or
services.
```

Additional resources

- Red Hat Ecosystem Catalog

6.2. CONFIGURING THE GPU THAT YOU WANT TO USE WITH THE COMMAND-LINE ASSISTANT

For better performance, you can change the configuration settings of the graphics processing unit (GPU) attached to the host system by editing the **~/.config/rhel-cla/.env** file. This file contains information to help you select the correct container image and device settings for your hardware.

Prerequisites

- You have installed the RHEL command-line assistant in a disconnected environment. See Installing the RHEL command-line assistant in disconnected environments .
- You ran the **rhel-cla start** command successfully at least once to generate the **~/.config/rhel-lightspeed/.env** file.

Procedure

1. Open the **~/.config/rhel-cla/.env** file in a text editor. For example:

   ```
   $ vi ~/.config/rhel-cla/.env
   ```

2. Set the **container image** variable:

   a. Locate the **LLAMACPP_IMAGE** variable and set it to the corresponding **ramalama** container for your GPU hardware. For example:

   ```
   LLAMACPP_IMAGE="quay.io/ramalama/cuda:latest"
   ```

3. Set the **host device** variable:

   a. Locate the **HOST_DEVICE** variable for the device of your GPU. For example:

   ```
   # For AMD GPUs
   HOST_DEVICE="/dev/dri"
   # For NVIDIA GPUs
   HOST_DEVICE="/dev/nvidia0"
   ```

4. Optional: Configure NVIDIA-specific variables:
   a. Locate and set up the NVIDIA-specific variables.
5. Restart the assistant to apply the changes:

```
$ rhel-cla stop
$ rhel-cla start
```

Troubleshooting

- Resolve GPU errors after restarting an EC2 instance.
  When you use the offline RHEL command-line assistant on a GPU-enabled AWS EC2 instance, the **rhel-cla start** command might fail after the instance is stopped and restarted. This error happens because the GPU ID changes after a restart. To work around this problem, regenerate the NVIDIA Container Device Interface (CDI) configuration file by running the following command:

```
$ nvidia-ctk cdi generate --output=/etc/cdi/nvidia.yaml
```

  This updates the reference of the system to the GPU, enabling the **rhel-cla** service to start correctly.

6.3. CHANGING THE LLM TO A CUSTOM MODEL

To use a custom model that you have installed locally, you must copy the custom model file into the **ramalama** LLM container and update your configuration.

Prerequisites

- The **rhel-cla** service has been started at least once by (rhel-cla start).

Procedure

1. Identify the running container ID:

```
$ podman ps | grep ramalama
6eef0fb344b7  quay.io/ramalama/ramalama:latest                              ramalama --
store ...  7 seconds ago  Up 7 seconds  0.0.0.0:8000->8000/tcp, 0.0.0.0:8888->8888/tcp
rhel-cla-llamacpp-model
```

2. Copy the LLM model file into the **ramalama** container by replacing *<path_to_local_model>* with your local path and *<container_id>* with the container ID found in the previous step:

```
$ podman cp path_to_local_llm_file.gguf 6eef0fb344b7:/models/
```

   Alternatively, you can copy the LLM model file directly to the local volume path:

```
$ cp path_to_local_llm_file.gguf ~/.local/share/containers/storage/volumes/llamacpp-
models/_data/
```

3. Stop the **rhel-cla** service:

```
$ rhel-cla stop
```

4. Edit the **~/.config/rhel-cla/.env** and add the following configuration to the **LLM** variable:

```
LLM="file:///models/llm_file.gguf"
```

5. Restart the **rhel-cla** service:

```
$ rhel-cla start
```

CHAPTER 7. USING THE COMMAND-LINE ASSISTANT TO DEBUG OR TROUBLESHOOT SYSTEM ISSUES

You can use the command-line assistant powered by RHEL Lightspeed to request information on how to troubleshoot the issues that you face on your system.

To troubleshoot your system, ask a question by using the following syntax: **c** + "*<question>*". For example:

- **$ c "how to troubleshoot network errors"**
- **$ c "I cannot access my server with SSH. Can you give me a list of things to troubleshoot?"**
- **$ c "I am failing to start sssd process"**
- **$ c "I need to boot into a different kernel"**
- **$ c "how to troubleshoot SSHD failing to start"**
- **$ c "how do I find all the files in the /etc that have been modified in the last hour"**
- **$ c "I am failing to start sssd process"**

7.1. USING THE COMMAND-LINE ASSISTANT TO TROUBLESHOOT THE SSHD SERVICE FAILING TO START

You can use the command-line assistant powered by RHEL Lightspeed to troubleshoot an SSHD service that fails to start.

You can use these command-line assistant features:

- An optional terminal capture feature in the command-line assistant to reference the output of previous commands when interacting with the command-line assistant.
- Piping data into the command-line assistant.

Prerequisites

- You have enabled the command-line assistant.
- You have root access to your system.

Procedure

1. Check the SSHD status and restart it.

```
$ sudo systemctl status ssh
$ sudo systemctl restart ssh
```

2. Enable the optional command-line assistant terminal capture feature:

```
$ c shell --enable-capture
```

3. Use the **-w 1 *"your_question"*** option to specify to include the output from the last command that was run.

   ```
   $ c -w 1 "what_is_this"
   ```

   - If you specify the number 2, that references the output from the second-to-last command. This is also true for the additional numbers.
   - You can also specify a prompt to run with the command and ask "help me understand the output", and reference the output with the error, so that the command-line assistant understands that you are asking for more details on what the error is.
     The command-line assistant processes the request and provides several possible solutions. In the example, you can use the suggestion to run the **journalctl -xeu sshd.service** command to check the log files.

4. Run that **journalctl** command and check the log files to identify potential issues.

   ```
   $ journalctl -xeu sshd.service
   ```

5. Ask the command-line assistant to generate a command on how to fix this typing error.

   ```
   $ c "what is the command that I can use to change 'Porrt' to 'Port' in the
   /etc/ssh/sshd_config file?"
   ```

6. Run the command suggested by the command-line assistant as a sudo user. For example:

   ```
   $ sudo sed -i s/Porrt/Port/g /etc/ssh/sshd_config
   ```

Verification

- Restart the SSHD service and check the status of the SSHD.

  ```
  $ sudo systemctl restart sshd
  $ sudo systemctl status sshd
  ```

7.2. USING THE COMMAND-LINE ASSISTANT TO TROUBLESHOOT SELINUX ISSUES

You can troubleshoot SELinux issues by using the command-line assistant. The example troubleshooting process demonstrates the diagnostic capabilities of the command-line assistant.

Prerequisites

- You have enabled the command-line assistant.
- You have root access to your system.

Procedure

1. On your terminal, enter the following command to list the **httpd** package version that you have installed in your system:

```
$ sudo rpm -qa httpd
httpd-2.4.62-2.fc40.x86_64
```

2. Query all **httpd** packages:

   ```
   $ sudo rpm -qa httpd
   ```

3. Identify the ports on which the web server accepts incoming requests:

   ```
   $ cat /etc/httpd/conf/httpd.conf | grep Listen
   # Listen: Allows you to bind Apache to specific IP addresses and/or
   # Change this to Listen on a specific IP address, but note that if
   #Listen 12.34.56.78:80
   Listen 80
   ```

4. Restart the **httpd** service:

   ```
   $ systemctl restart httpd

   Job for httpd. Service failed because the control process exited with error code.
   See "systemctl status httpd.service" and "journalctl -xeu httpd.service" for details.
   ```

   a. Run the **journalctl** command for more details on the failed service:

      ```
      $ sudo journalctl -xeu httpd.service
      ```

5. Use the command-line assistant to troubleshoot the issue and ask why the service is failing:

   ```
   $ sudo c "why did httpd fail to start"
   ```

   a. Ask the command-line assistant about the **selinux httpd** port:

      ```
      $ c "selinux httpd port"
      ```

      The assistant advises using the **sestatus** command to check the current SELinux status and the content of the httpd services with the following command:

      ```
      $ sudo sestatus

      SELinux status:                 enabled
      SELinuxfs mount:                /sys/fs/selinux
      SELinux root directory:         /etc/selinux
      Loaded policy name:             targeted
      Current mode:                   enforcing
      Mode from config file:          enforcing
      Policy MLS status:              enabled
      Policy deny_unknown status:     allowed
      Memory protection checking:     actual (secure)
      Max kernel policy version:      33
      ```

   b. View the specific SELinux policy for the httpd services by running the following command:

```
$ sudo cat /usr/share/selinux/targeted/contexts/httpd_var_run_t
No such file or directory
```

c. Ask the command-line assistant about contexts.

```
$ c "i don't have a httpd_var_run_t contexts"
```

The command-line assistant takes some time to process the request, then provides several possible suggestions.

d. The assistant says that you might not have context and need to set it with the following command:

```
$ sudo chcon -R -t httpd_var_run_t
```

e. Ask the CLA about the port:

```
$ c "selinux won't let httpd listen on port 12345"
```

f. Try the following suggestion, run the command:

```
$ sudo semanage port -a -t httpd_port_t -p tcp 12345
ValueError: Type httpd_port_t is invalid, must be a port type
```

g. Ask the CLA about the error you see in the output:

```
$ c "how do I fix ValueError: Type httpd_port_t is invalid, must be a port type"
```

6. Run the steps provided by the CLA:

```
$ sudo ls -Z /usr/sbin/httpd
system_u:object_r:httpd_exec_t:s0 /usr/sbin/httpd

$ chcon -t httpd_exec_t /usr/sbin/httpd

$ sudo setenforce 1
```

a. Restart the **httpd** service and check the status of **httpd.service**:

```
$ sudo systemctl restart httpd
Job failed

$ sudo systemctl status httpd.service
Failed to start the Apache Server
```

7. Ask the CLA how to enable **httpd** to listen on **port** 12345:

```
$ c "how do I enable httpd to listen on port 12345 selinux"
```

a. Run the command advised by the CLA:

```
$ sudo setsebool -P httpd_can_network_connect=1
```

8. Restart the **httpd** service and check the status of **httpd.service** again:

   ```
   $ sudo systemctl status httpd
   $ sudo systemctl restart httpd
   Job failed, see journalctl
   ```

9. Check the **journalctl** service:

   ```
   $ journalctl -xeu httpd
   Output: An ExecStart= process belonging to unit httpd.service has exited.
   ```

10. Use the output to ask the CLA to troubleshoot:

    ```
    $ c "An ExecStart= process belonging to unit httpd.service has exited."
    ```

    a. Run the command that the CLA responds with:

    ```
    $ sudo ausearch -m AVC,USER_AVC -ts recent
    Output: "avc:  denied  {name_bind}  for pid=7184 comm="httpd" src=12345
    scontext=system_u:system_r:httpd_t:s0 tcontext=system_u:object_r:
    unreserved_port_t:s0 tclas=tcp_socket permissive=0"
    ```

    b. Copy the output of the previous command:

    ```
    $ sudo c "avc: denied {name_bind} for pid=7184 comm="httpd" src=12345
    scontext=system_u:system_r:httpd_t:s0 tcontext=system_u:object_r:
    unreserved_port_t:s0 tclas=tcp_socket permissive=0"
    ```

    c. Run the following command to resolve the error “SELinux is preventing Apache Server (httpd) from binding to port 12345".

    ```
    $ sudo semanage port -a -t http_port_t -p tcp 12345
    ```

Verification

- Restart the httpd service and check the status of **httpd.service**:

  ```
  $ sudo systemctl restart httpd
  No error
  $ sudo systemctl status httpd.service
  ```

  The server is configured, up and running, and listening on **port 443**, **port 12345**.

CHAPTER 8. TROUBLESHOOTING THE COMMAND-LINE ASSISTANT POWERED BY RHEL LIGHTSPEED

Your system configuration might cause issues when installing and using the command-line assistant powered by RHEL Lightspeed. The following sections provide instructions on fixing the most common problems.

8.1. SOLVING INABILITY TO FIND THE TLS CERTIFICATE FILE ERROR

When you run the **c** command on your system, it might fail with the following error: "Could not find the TLS certificate file, invalid path: */etc/pki/consumer/cert.pem*".

This error happens when the command-line assistant cannot find the TLS certificate file because of an invalid path. This could mean that your system is not registered.

Prerequisites

- You have installed the command-line assistant powered by RHEL Lightspeed.

Procedure

1. Check if the system is registered:

```
# subscription-manager identity
```

2. If the system is not registered, use the following command to register the system:

```
# subscription-manager register
```

8.2. SOLVING COMMUNICATION ERRORS WITH THE SERVER

If the **c** command fails when you run it on your system, it might fail due to a communication error with the server. To solve this issue, restart **clad**.

```
  Communication error with the server:
HTTPSConnectionPool(host='cert.console.redhat.com', port=443): Max retries exceeded with
url: /api/lightspeed/v1/infer (Caused by ProtocolError('Connection aborted.',
PermissionError(13, 'Permission denied'))). Please try again in a few minutes.
```

To fix the failing command, solve the communication error with the server.

Prerequisites

- You have installed the command-line assistant powered by RHEL Lightspeed.

Procedure

1. Restart the command-line assistant daemon (**clad**).

```
$ systemctl restart clad
```

2. Rerun the **c** command with your question:

```
$ c "<Your_question>"
```

8.3. THE c COMMAND WARNS ABOUT THE SIZE OF AN ATTACHMENT

When you run the **c** command and try to attach a file, you might receive a warning about the size limit.

This happens because the assistant has a 2KB limit for the client. Starting with RHEL 10.1, the context limit is 32KB.

```
"Error: The total size of your question and context (478.46 KB) exceeds the limit of 2.00 KB.
Trimming it down to fit in the expected size, you may lose some context."
```

If a file exceeds the maximum size, the system truncates the file, which can result in incomplete data or loss of context.

8.4. RELOADING THE UPDATED CERTIFICATES

If you run the command-line assistant daemon (**clad**) as a non-root user, the updated certificates do not reload correctly. This issue occurs because only root users can own and access the certificate files located in the **/etc/pki/consumer/** directory.

To fix this issue, use the following steps:

Prerequisites

- You have installed the command-line assistant powered by RHEL Lightspeed.

Procedure

1. Change the **key.pem** permission to root.

```
$ sudo ls -l /etc/pki/consumer/
$ sudo chown $(whoami):$(id -gn) /etc/pki/consumer/${<key_name>}
```

2. Run **clad** commands as root.

```
$ sudo chown $(whoami):$(id -gn) /etc/pki/consumer/${<key_name>}
```

CHAPTER 9. MODIFYING THE CONFIGURATION OF THE COMMAND-LINE ASSISTANT

The command-line assistant daemon (**clad**) is the core of the command-line assistant powered by RHEL Lightspeed. **clad** manages communication with Red Hat Lightspeed services, such as user history management.

Optionally, to integrate the Red Hat Lightspeed services with your existing infrastructure, you can modify the configuration of the command-line assistant to use a network proxy or connect to a different database.

9.1. SETTING UP A PROXY CONFIGURATION

If you need a proxy for Internet access, you can set up a proxy configuration by making the following changes in the **config.toml** configuration file.

Prerequisites

- The command-line assistant powered by RHEL Lightspeed is installed.

Procedure

1. Access the proxy configuration by opening the **/etc/xdg/command-line-assistant/config.toml** configuration file.
2. Locate and change the following block in the **config.toml** file :

```
# Backend settings for communicating with the external API.
[backend]
...
# proxies = { http = "http://example-host:8002", https = "https://example-host:8002" }
```

3. Uncomment the **proxies** key and define your **http** or **https** proxy host:

```
[backend]
…
# For a https proxy host
proxies = { https = "https://<your_https_proxy_host:1234>"}
```

4. After making the changes, restart **clad** for the changes to be effective:

```
$ sudo systemctl restart clad
```

NOTE

You can use the **http** value and **https** key control if the http or https traffic from **clad** is routed to the specified proxy. However, the protocol does not influence the proxy type selection, and you can have a configuration that uses http proxy for https traffic. For example:

```
https = "http://<your_https_proxy_host:1234>"
```

Additional resources

- urllib.request extensible library for opening URLs documentation

CHAPTER 10. MANAGING DATABASES WITH THE COMMAND-LINE ASSISTANT DAEMON

To store information and provide access to your history database, the command-line assistant daemon (**clad**) uses an unencrypted SQLite database by default. If you require a different database for your deployment, you can install and connect to a different database back end, such as PostgreSQL or MySQL.

NOTE

clad does not include these databases by default to avoid bringing unwanted dependencies to your system.

10.1. CHANGING THE DEFAULT DATABASE IN THE CONFIGURATION FILE

With the unencrypted SQLite database, you can store information and have access to your history database from the command-line assistant.

Prerequisites

- You have installed the command-line assistant.

Procedure

1. Install the database of your choice:
   - To install MySQL, enter:

     ```
     # dnf install python3-PyMySQL
     ```

   - To install PostgreSQL, enter:

     ```
     # dnf install python3-psycopg2
     ```

2. Access your database configuration file at **/etc/xdg/command-line-assistant/config.toml**.
3. Locate and comment out the default configuration. For example:

   ```
   [database]
     # type = "sqlite"
     # connection_string = "/var/lib/command-line-assistant/history.db"
   ```

4. Configure the database of your choice. The following information is also available in **/etc/xdg/command-line-assistant/config.toml**.
   a. Set the database type, where ***<db_type>*** can be **mysql** or **postgresql**.
   b. Set the database details.

   ```
   type = <db_type>
   host = "<hostname_or_ip_address>"
   ```

```
port = "5432"
username = "<database_user_name>"
password = "<password>"
database = "<database_name>"
```

5. After changing the database type, restart the **clad** daemon to apply the changes:

```
$ sudo systemctl restart clad
```

10.2. CONNECTING TO YOUR DATABASE BY USING THE STORED SYSTEMD-CREDS PASSWORDS

You can use the **systemd-creds** tool to securely store your encrypted credentials, and use the secrets stored in **systemd-creds** to connect to the database of your choice: PostgreSQL, SQLite, or MySQL.

Prerequisites

- The command-line assistant.
- Access to the database configuration file.

Procedure

1. Access your database configuration file at **/etc/xdg/command-line-assistant/config.toml**.
2. Remove the **username** and **password** parameters from the **[database]** section, for example:

```
[database]
type = "postgresql"
host = "localhost"
port = "5432"
database = "history"
```

NOTE

If you leave the username and password in the configuration file, these credentials take precedence over the **systemd-creds** tool.

3. Generate encrypted credentials for your username or password. The following example uses **systemd-ask-password** commands. The name must follow the schema of **database-username** and **database-password**, otherwise, **clad** does not load the credentials properly.

   a. To generate an encrypted username, run the following command:

```
$ systemd-ask-password -n | ( echo "[Service]" && systemd-creds encrypt --
name=<database_username> -p - - )
>/etc/systemd/system/clad.service.d/<username>.conf
```

   b. To generate an encrypted password, run the following command:

```
$ systemd-ask-password -n | ( echo "[Service]" && systemd-creds encrypt --
name=<database_password> -p - - )
>/etc/systemd/system/clad.service.d/<password>.conf
```

4. After updating the database credentials, reload **systemd** and restart the **clad** daemon to apply the changes:

   ```
   $ sudo systemctl daemon-reload
   $ sudo systemctl restart clad
   ```

CHAPTER 11. USING THE RHEL MCP SERVER TO ENABLE AI ASSISTANTS TO RUN, DISCOVER, AND TROUBLESHOOT COMPLEX ISSUES

Using the RHEL Model Context Protocol (MCP) server, you can enable LLMs to act as system administrators, capable of discovering and troubleshooting complex issues across your infrastructure.

You can use an AI assistant of your preference, such as Goose or Gemini CLI, and by using SSH through your own account, the tools that your AI assistant runs are subject to the same security restrictions. The MCP server provides a protected way for AI models to perform system administration tasks, troubleshoot issues, and read configuration files on the target system machines.

Use the RHEL MCP server for read-only Linux system administration with an agent-based AI of your choice on RHEL 10 Base OS, on-premise hosts. The RHEL MCP server is available through the RHEL Developer Subscription for production testing and can run on RHEL or Linux, macOS, and Windows 11 operating systems.

IMPORTANT

Red Hat Enterprise Linux (RHEL) command-line assistant is Developer Preview software only. Developer Preview software is not supported by Red Hat in any way and is not functionally complete or production-ready. Do not use Developer Preview software for production or business-critical workloads. Developer Preview software provides early access to upcoming product software in advance of its possible inclusion in a Red Hat product offering. Customers can use this software to test functionality and provide feedback during the development process. This software might not have any documentation, is subject to change or removal at any time, and has received limited testing. Red Hat might provide ways to submit feedback on Developer Preview software without an associated SLA.

For more information about the support scope of Red Hat Developer Preview software, see Developer Preview Support Scope.

The RHEL MCP server has the following capabilities:

- Inspect local and target systems for troubleshooting.
- Public key discovery to list available public keys from the local **~/.ssh** directory.
- Comprehensive error handling with clear error messages for connection and execution issues.

The MCP server has the following system management features:

- Read-only operations: All tools are strictly read-only for safe diagnostics.
- Remote SSH execution: Inspect systems by using SSH with key-based authentication.
- Local execution: Inspect the local system (when the MCP server is not running in a container).
- Multi-host management: Connect to different target system hosts in the same LLM session.
- Comprehensive diagnostics: System information, services, processes, logs, network, and storage.
- Configurable log access: Use environment variables to control which log files are accessible.

11.1. USING SSH TO AUTHENTICATE THE RHEL MCP SERVER TO A TARGET MACHINE

To enable the RHEL MCP server to connect to your target machine by using SSH, you must use an account with key-based authentication.

Prerequisites

- A client machine with the operating system of your choice.
- A RHEL target machine.
- SSH connectivity between the host running the MCP server and the RHEL target machine.

Procedure

1. On the client machine, generate an SSH keypair:

   ```
   $ ssh-keygen -t ed25519 -C "<your-email@example.com>" -f id_ed25519_mcp
   ```

2. On the RHEL target machine, create a non-root user:

   a. Access the VM by SSH and run the following commands:

   ```
   $ sudo useradd mcp
   $ sudo passwd tester
   ```

3. On the client machine, add the target machine to your SSH configuration file:

   a. Edit the **~/.ssh/config** configuration file and add the following information:

   ```
   Host <rhel-10-0>
   HostName <target-machine-ip-address>
   User <user-test>
   Port 22
   IdentityFile <path-to-your-ssh-pub-key>
   StrictHostKeyChecking no
   ```

4. Copy your public key from your client machine to the target machine.

   ```
   $ ssh-copy-id -i ~/.ssh/id_ed25519_mcp.pub mcp@<vm_ip_address>
   ```

5. Restart the SSH service:

   ```
   $ sudo systemctl restart sshd
   ```

11.2. INSTALLING THE RHEL MCP SERVER

You can install the RHEL MCP server on your host to use it with any AI client. Choose one of the following methods to perform the installation based on your security requirements.

- Running the MCP server from a container for enhanced security.

- Installing the MCP server locally by using **pip** package manager for direct host administration.

11.2.1. Running the RHEL MCP server from a container image

You can run the MCP server on your system by using a container image. The MCP server uses SSH to connect to target system hosts. You must make the SSH keys available inside the container. If the SSH key is encrypted, you must provide a passphrase to decrypt the key.

NOTE

In container-based scenarios, you cannot use the RHEL MCP server to inspect the local system.

Prerequisites

- The **container-tools meta-package** is installed.
- Podman is installed on your host machine.
- A target machine where the RHEL MCP server executes commands by SSH.

Procedure

- On the LLM client, run the container image:

```
podman run --rm --interactive --userns keep-id:uid=1001,gid=0 -e
LINUX_MCP_KEY_PASSPHRASE -e LINUX_MCP_USER -v
$HOME/.ssh/id_ed25519_mcp:/var/lib/mcp/.ssh/id_ed25519:ro,Z -v
$HOME/.local/share/linux-mcp-server/logs:/var/lib/mcp/.local/share/linux-mcp-
server/logs:rw,Z quay.io/redhat-services-prod/rhel-lightspeed-tenant/linux-mcp-server:latest
```

The **--rm** option removes the container image after the container exits.

Verification

- List the running containers:

```
$ podman ps
```

11.2.2. Installing the RHEL MCP server by using pip

The RHEL MCP server runs locally on the same machine as your AI agent. The RHEL MCP server is responsible for connecting to your target RHEL machine. Your AI agent communicates with this local server by standard I/O (**stdio**).

Prerequisites

- Python 3.10 or higher.

```
$ sudo dnf install python3 python3-pip
```

Procedure

- Install the package from PyPI:

  ```
  $ pip install --user linux-mcp-server
  ```

Verification

- Test the server:

  ```
  $ ~/.local/bin/linux-mcp-server
  ```

  The server starts and displays initialization messages.

- To stop the MCP server, press the **Ctrl+C** keys, then the **Return** key.

Next steps

- Configure the MCP server for use with your AI client .

11.2.3. Configuring the RHEL MCP server in your AI client

To integrate your AI clients with the RHEL MCP server, use a JSON configuration and embed it directly into the AI application settings file or include it as a separate JSON file.

The RHEL MCP server JSON configuration requires the following parameters:

MCP Server or Connection Type	Standard IO (STDIO)
ID / Name	**linux-tools**
Description	Linux system diagnostics
Command	**/home/*<your-username>*/.local/bin/linux-mcp-server**
Arguments	[] (*Empty array*)
Environment Variables	**LINUX_MCP_USER=*<your-ssh-username>***

Prerequisites

- AI client of your choice.

Procedure

- Add the **mcpServers** JSON configuration to integrate into your AI application:

  ```
  {
   "mcpServers": {
     "linux-mcp-server": {
       "command": "~/.local/bin/linux-mcp-server",
       "args": [],
  ```

```
      "env": {
        "LINUX_MCP_USER": "<your-ssh-username>"
      }
    }
  }
}
```

11.3. USING THE RHEL MCP SERVER TO QUERY INFORMATION FROM A RHEL SYSTEM

You can use the RHEL MCP server to enable an LLM to obtain and analyze RHEL system logs, including **journald** and **syslog** data, to detect advanced anomalies, security threat identification, and AI-driven root cause analysis, turning log data into actionable intelligence.

You can also use the RHEL MCP server to enable an LLM to run commands on the RHEL system.

Prerequisites

- You installed the RHEL MCP server.

Procedure

1. Optional: Configure environment variables to select which log files the RHEL MCP server can access, and the logging level required.

   ```
   export
   LINUX_MCP_ALLOWED_LOG_PATHS="/var/log/messages,/var/log/secure,/var/log/audit/audit
   .log"
   export LINUX_MCP_LOG_LEVEL="INFO"
   ```

2. Run the RHEL MCP server.

   ```
   $ linux-mcp-server
   ```

3. Ask questions, for example:

   ```
   "What version of RHEL is this system running?"
   "Show me the current CPU load and usage per core."
   "How much memory is being used on this system?"
   "Which filesystems are running out of space?"
   "What hardware is installed in this system?"
   "Show me all critical and error logs since yesterday."
   ```

11.4. TROUBLESHOOTING THE RHEL MCP INSTALLATION ISSUES

Current known issues and limitations.

- You might experience issues getting the SSH authentication working in the container, especially with Docker, because it does not support user namespace mapping. To workaround this issue, the SSH key file must be owned by UID 1001 on the container host.
- When you run the RHEL MCP server in a container, it cannot inspect the local system due to the isolation of containerized environments.

CHAPTER 12. FREQUENTLY ASKED QUESTIONS ABOUT USER DATA SECURITY

Review the following information about how the command-line assistant uses your data.

What system information can the command-line assistant directly access?

The command-line assistant powered by RHEL Lightspeed cannot directly access information from your system. For example, if you ask "how much free memory does this system have?", the command-line assistant cannot directly gather this information from your system. Instead, the command-line assistant can help you find commands to display how much free memory the system has.
You can provide information to the command-line assistant by using various methods, such as the following:

- Including the information in the prompt or question.
- Attaching a file to the command-line assistant with the **--attachment** option.
- By using a shell pipeline to pass output from a command to the command-line assistant.
- By using the **--with-output** option.

What is the process flow for input and output in the command-line assistant?

1. The command-line assistant receives your question as an input.
   a. The command-line assistant logs and stores complete transcripts of conversations with the user. This includes the following information:
      - Queries from the user.
      - The complete message is sent to the configured Large Language Model (LLM) provider, which includes system instructions, referenced documentation, and the user question.
      - The complete response from the LLM provider.
2. The input is processed by the back end.
3. The command-line assistant searches for relevant knowledge related to your query.
4. The command-line assistant takes your query, the relevant knowledge, and other instructions, and sends everything for AI inference.
5. You receive an output from the command-line assistant as a response.
   The command-line assistant is not intended to process personal information, and you agree to not include any personal information in the input.

   Note that the interactions that you have with the command-line assistant are logged locally on your system, which makes it possible for you to access your chat history. These interactions are also logged on the service, and may be used to improve Red Hat’s products or services.

Additional resources

- Attaching a file to your questions to the command-line assistant
- Redirecting a command output to the command-line assistant