Red Hat Enterprise Linux 9 – Full Walkthrough

Pre installation

Download RHEL 9 ISO

From Red Hat Customer Portal or Developer site. You'll need a subscription (trial available for developers).

Create bootable USB/DVD

Use a tool like Rufus (Windows) to burn the ISO.

Prepare ASUS NUC hardware (any other choice of hardware will be fine as long as it supports RHEL 9x)

- Connect monitor, keyboard, mouse, and Ethernet cable (DHCP recommended).
- Check & update BIOS/UEFI firmware.
- Plug in the bootable USB.
- Enter BIOS (F2/DEL key at boot).
- Set boot device → USB.
- Enable Intel VT-x/VT d
- Save and exit
- Go to the command prompt
- Format nvme
  - lsblk /dev/nvme0n1
  - sudo umount /dev/nvme0n1p1
  - sudo umount /dev/nvme0n1p2
  - # Repeat for all listed partitions
  - sudo wipefs -a /dev/nvme0n1
- Reboot.
- Boot from USB

Installation

Boot from USB → Select Install Red Hat Enterprise Linux 9.

Installer setup steps:

- Choose language and confirm.
- Set time zone (Date & Time menu).

- Configure regional settings (Keyboard, Language).
- Installation Destination → pick internal NVMe/SSD.
- Software selection → pick Server Without GUI. Add:
    - Debugging tools
    - Performance tools
    - Hardware monitoring
    - Remote management for Linux
    - Security tools
    - Container management tools
    - Console internet tools
    - Headless management tools
    - System tools
- System purpose & support plan → Configure role & SLA if it's for production.
- Networking → Enable NIC (defaults to DHCP) and optionally configure hostname.
- Set root password
- Create a user account and assign a password
- Security Policies (optional).
- Review settings → Click Begin Installation.

Wait, then click Reboot when done. Remove USB.

Post installation setup

Now login to the new system, the recommendation is to do it remotely via ssh.

Register system to Red Hat (subscription-manager register) if not done before.

Update system immediately:

```
sudo dnf update -y
```

Remote Access Hardening

SSH key setup from your admin machine:

```
ssh-keygen -t rsa -b 4096
```

```
ssh-copy-id <user>@<nuc_ip>
```

Now run edit this file on the newly installed system /etc/ssh/sshd_config:

```
PermitRootLogin no
PubkeyAuthentication yes
PasswordAuthentication no
ChallengeResponseAuthentication no
```

Restart SSH:

```
sudo systemctl restart sshd
```

Log in via SSH using your key.

Final Hardening & Tools

Configure firewalld rules depending on the server role.

Run:

```
systemctl start irqbalance.service
```

```
insights-client
```

You should now be set to deploy our sensor using the container we provide. Move on to the next document where you will get the necessary steps to do just that.

Steps to deploy a single container on RHEL 9

The following are the steps to configure the environment and get all the requirements in place to deploy the container image. These steps and the GuardDog AI sensor have been tested with RHEL versions 9.5, 9.6. This guide assumes RHEL is already installed, configured and ready to deploy the GuardDog AI container.

Prerequisites, Assumptions and Additional Configuration

- Network traffic visibility: Port mirroring, for both ingress and egress, is preferred and necessary. SPAN ports may also be utilized but additional configuration and installation changes are needed. We need to be able to see all packets on the network, and to communicate and transmit out on the same network.
- DNS Configuration: Necessary if you encounter problems downloading container images.
- Network Configurations: Make the necessary adjustments for effective communication within your network. Firewall rules are crucial to allow communication back and forth between the cloud and this container, please make sure you follow those and make the necessary changes before attempting to start the container.
- Host network interfaces: The container is prepared to work with the network interfaces presented to the host. Have those configured before running the container (vlans, logical, physical).

- ➢ Container runs with host networking access to be able to be completely functional. It will detect all ethernet interfaces and add them to the internal network configuration within the container.
- ➢ It uses a persistent volume in order to be able to maintain its configuration throughout container and image updates and reboots.
- ➢ A license is needed to run the container, and this will be provided as part of the deployment process.

The container is meant to be run following this guide utilizing the restart switch in the command to be able to have the container come back up when rebooting the host system. Follow the instructions below.

This first set of requirements is to create an account in our system to get things started. Once this first section is completed, we will send an email with the necessary parameters and variables to run the container.

Create an account

- Create an account at https://dcx.guarddog.ai
- Click on Sign Up
- Fill out the short online form
- Verify your account
- Login with your account to fully activate it
- Contact your sales representative and/or support to let us know you have completed this step and the email address you used for the account.

Once the account has been created, please send us an email to support@guarddog.ai to let us know you are ready to proceed to the next step.

DNS Configuration for the container on RHEL

In some cases, we have seen issues with DNS not being properly configured. This is the case all the time, but we wanted to include this additional step in case it is needed.

Disable DNS Management by NetworkManager

```
echo -e "[main]\ndns=none" | sudo tee /etc/NetworkManager/conf.d/90-dns-none.conf
&& sudo systemctl reload NetworkManager
```

Manually Set DNS Servers

```
echo -e "nameserver 8.8.8.8\nnameserver 8.8.4.4" | sudo tee -a /etc/resolv.conf
```

Disable Consistent Network Device Naming in RHEL

We have seen some strange behavior when the naming for the network interfaces naming is not configured correctly. This method allows us to use the standard naming convention. This is not mandatory and is only needed if issues are seen when running the container and when suggested by our engineering team.

- Edit Kernel Parameters in GRUB

```
sudo sed -i 's/GRUB_CMDLINE_LINUX="\(.*\)"/GRUB_CMDLINE_LINUX="\1
net.ifnames=0 biosdevname=0"/' /etc/default/grub
```

- Regenerate GRUB Configuration
  - For BIOS systems:

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

  - For UEFI systems:

```
grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg
```

- System Reboot

```
reboot
```

Container Deployment

This step goes over the process of getting the latest container image downloaded from Docker Hub and then the command needed to start the container. This command includes the license, and some other parameters explained below.

Download the container Image

```
podman pull docker.io/guarddogai/prod:latest
```

By now you should have received an email from us with the parameters needed for the run command below. In case you have not, please follow up with support.

Here is the command to start the container. Please make sure you follow the instructions to provide the right information when issuing the command.

The recommended way to manage auto-restarting containers in Podman, especially on a system like RHEL, is by using systemd services. This is because Podman is daemonless, and integrating with systemd provides more robust service management.

Create and run the container initially without the --restart flag. Use -d to run it in the background (detached).

```
podman run -it -user root --cap-add NET_ADMIN -cap-add NET_RAW --net=host -v
/etc/$DEVICE_NAME:/etc/guarddog:Z --name $DEVICE_NAME docker.io/guarddogai/prod:latest gdai --
device_name=$DEVICE_NAME --email=$EMAIL --license=$LICENSE
```

Generate a systemd unit file for the container.

```
podman generate systemd --new --name $DEVICE_NAME > /etc/systemd/system/container-
$DEVICE_NAME.service
```

Reload the systemd daemon, enable, and start the new service.

```
# Reload systemd to recognize the new service file
systemctl daemon-reload
# Enable the service to start automatically on boot
systemctl enable container-$DEVICE_NAME.service
# Start the service immediately
systemctl start container-$DEVICE_NAME.service
```

<DEVICE_NAME>, indicates a friendly name for the container. This will show in the user interface to locate and identify the running container. This is assigned by the client and can be any name to make it easier to remember and organize deployments.

<EMAIL>, this is the email address used to create the account at https://dcx.guarddog.ai.

<LICENSE>, this is the license key that will be provided by GuardDog AI for each container to be deployed. The license is not transferable or cannot be interchangeably used with other containers. Therefore, each license is unique to 1 container.

This is all you need to do. Given that you had the network setup correctly following the firewall document you should have received, the container will start up and go through a series of checks to make sure it is able to communicate with the cloud. Then it will proceed to validate the license and, if valid, then to register with the system. The container is set up so that it automatically registers to your account in our system. Please allow for sufficient time to let the container complete the entire process. Although it is usually done within a few minutes, depending on how busy the network is and the bandwidth available, it may take several hours.