SUSE

Introduction to Ansible Core

WHAT?

Ansible is an IT automation tool that simplifies configuration management, application deployment and task orchestration by enabling you to define infrastructure as code in a simple way.

WHY?

Learn how to automate IT infrastructure with Ansible, from installation and configuration, to creating inventory files and a playbooks.

EFFORT

The average reading time of this article is approximately 40 minutes.

REQUIREMENTS

- *Linux fundamentals:* Understanding basic Linux commands, file permissions, directory structures and use of the command line.
- *Networking:* Ansible connects to remote machines via SSH so knowledge of IP addresses, SSH, host names and ports is required.
- *YAML:* Ansible playbooks are written in YAML, so knowing how to structure a YAML file is essential.

Publication Date: 07 Aug 2026

Contents

1 What is Ansible core?

Ansible core is an open source IT automation engine that automates provisioning, configuration management, application deployment and other IT processes. It is has agent less architecture, which means no dedicated connection software is required on the managed nodes. It operates on the following concepts:

- *Control node:* The machine where Ansible is installed and executed.
- *Managed nodes:* The target servers or devices that Ansible manages.
- *Inventory:* A file that defines and groups your managed nodes.
- *Module:* A small program that performs actions on a managed node.
- *Task:* A single execution of an Ansible module on a managed node.
- *Playbooks:* Ansible playbooks provide a repeatable, reusable, simple configuration management and multi-machine deployment system.
- *Collection:* A collection of Ansible roles or modules focusing on a specific area.
- *Role:* A standard, self-contained and portable unit of automation that organizes related tasks, variables and files into a predictable directory structure for easy reuse.

Important

- **Ansible Core** is the foundational software package that includes command-line tools like `ansible` and `ansible-playbook` and the execution engine. It provides the essential libraries and basic modules needed to connect to managed nodes and execute tasks.
- **Ansible automation components** are the content that defines the logic, packaged into playbooks, roles and collections. These assets provide the instructions for the core engine, with collections serving as a distribution format that bundles reusable roles, modules and plug-ins.
- **Ansible Orchestration** is handled by higher-level tools that provide a UI, API and workflow engine to centrally manage and visually chain multiple automation playbooks together for complex multi-system processes.

2 Installing Ansible

Ansible is available for installation from SLES 16.0 base system.

1. Refresh the repository and install Ansible.

```
> sudo zypper refresh
```

```
> sudo zypper install ansible
```

2. To verify:

```
> ansible --version
  ansible [core 2.18.3]
  config file = None
  configured module search path = ['/home/amrita/.ansible/plugins/modules', '/usr/
share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3.13/site-packages/ansible
  ansible collection location = /home/amrita/.ansible/collections:/usr/share/
ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.13.5 (main, Jun 12 2025, 00:40:24) [GCC] (/usr/bin/python3.13)
  jinja version = 3.1.6
  libyaml = True
```

3 Understanding Ansible configuration

Ansible supports several sources for configuring its behavior, including an `INI` file named `ansible.cfg`, environment variables, command-line options and playbook keywords. The standard configuration is usually sufficient for most users.

The `ansible-config` utility provides users with a comprehensive view of all available configuration settings, including their default values, instructions on how to set them and the origin of their current values. Ansible processes the following list and uses the first file found. All others are ignored.

- `ANSIBLE_CONFIG` environment variable if set
- `ansible.cfg` located in the current directory
- `~/.ansible.cfg` located in the home directory
- `/etc/ansible/ansible.cfg`

You can generate a sample `ansible.cfg` file either with a fully commented out file or a more complete file that includes the existing plug-in. For example:

```
> sudo  ansible-config init --disabled > ansible.cfg
```

```
> sudo  ansible-config init --disabled -t all > ansible.cfg
```

4 Structure of an inventory file

An Ansible inventory file defines the hosts or servers on which Ansible commands and playbooks will run. It can be in either `INI` or `YAML` format and is a fundamental component of an Ansible project. An Ansible project is a directory structure containing all the files necessary to automate a specific IT goal, such as deploying an application or configuring a multi-tier infrastructure.

Ansible uses several built-in variables to connect to and manage hosts, which are often defined in the inventory file. The most common ones are:

- `ansible_host:` Specifies the IP address or DNS name of the host.
- `ansible_user` The user account to use for SSH connections.

- `ansible_port:` The connection port number if not the default, `22` for SSH.
- `ansible_private_key_file:` Specifies the path to the SSH private key file that Ansible should use to authenticate with and connect to the remote managed nodes.
- `ansible_python_interpreter:` Specifies the path to the Python interpreter on the remote host if it is not the default.

4.1 `INI` format

The `INI` format is the more traditional and widely used format for inventory files. Hosts can be listed individually or grouped together.

Individual hosts

You can list either the IP addresses or host names. For example:

```
host1.example.com
192.168.1.50
```

Groups

Groups are defined using `[]` brackets. You can apply tasks to multiple hosts simultaneously. A host can belong to multiple groups. For example:

```
[webservers]
web1.example.com
web2.example.com

[databases]
db1.example.com
```

Group variables

Variables can be assigned to all hosts within a specific group. These are defined under the group name using a `:vars` suffix. For example:

```
[webservers:vars]
ansible_user=test
http_port=80
```

Child groups

You can create a group of groups, known as a parent group, using the `:children` suffix. This is useful for combining different host types for a single task. For example:

```
[all_servers:children]
```

```
webservers
databases
```

4.2 YAML format

YAML offers a structured and hierarchical way to represent the inventory. It is often preferred for its readability and ability to handle complex data structures.

Host and group structure:

The inventory is defined under the `all` keyword, which contains hosts and children. For example,

```
all:
  hosts:
    host1.example.com:
    192.168.1.50:
  children:
    webservers:
      hosts:
        web1.example.com:
        web2.example.com:
    databases:
      hosts:
        db1.example.com:
```

Group and host variables

Variables are defined under a `vars` dictionary within the host or group. For example:

```
  webservers:
    hosts:
      web1.example.com:
      web2.example.com:
    vars:
     http_port: 80
     server_type: nginx
     max_connections: 500
```

Child groups

You can create a group of groups, known as a parent group, using the `:children` suffix. This is useful for combining different host types for a single task. For example:

```
all:
  children:
    webservers:
      hosts:
```

```
        web1.example.com:
          ansible_user: test
      vars:
        http_port: 80
```

5 Basic usage of Ansible

Ansible provides ad hoc commands and several utilities to perform various operation and automation tasks. Ad hoc commands in Ansible are useful for performing quick and one-off tasks that do not need to be saved for future use. They are single-line commands executed directly from the command line. Ad hoc commands use the `ansible` CLI tool.

In contrast, playbooks are a more powerful and scalable solution for defining complex, multi-step IT automation tasks. They are written in YAML and contain a list of plays, with each play associating a group of hosts with a series of tasks. The `ansible-playbook` CLI tool is used to execute these playbooks. This approach promotes reusability, version control and collaboration, making it the preferred method for managing and deploying applications at scale.

You can use either ad hoc commands or create a playbook to automate your IT infrastructure based on your use case.

5.1 Ad hoc commands

Ad-hoc commands are a quick way to run a single task on one or more managed hosts directly from the command line. Ideal for simple operations like checking system uptime or rebooting a server.

They are useful for quickly managing servers by performing actions like rebooting, copying files and managing packages and users. You can use any of the available Ansible modules in an ad hoc task. Like playbooks, ad hoc tasks use a *declarative* model, where you define the desired final state. Ansible then automatically calculates and executes the necessary steps to achieve that state. This approach ensures the task can be run multiple times without causing unintended changes, as Ansible first checks the current state of the host and only performs an action if it differs from the specified final state. Some examples include:

```
> sudo ansible webservers -m ansible.builtin.service -a "name=httpd state=started"
```

```
> sudo ansible all -m ansible.builtin.setup
```

5.2 Ansible command-line tools

Ansible provides the following command-line tools:

`ansible`

A simple tool or framework for remote tasks; this command-line tool allows you to define and run a single task playbook against a set of hosts.

```
> ansible -h
            usage: ansible [-h] [--version] [-v] [-b] [--become-method
 BECOME_METHOD] [--become-user BECOME_USER] [-K | --become-password-file
 BECOME_PASSWORD_FILE] [-i INVENTORY] [--list-hosts]
                           [-l SUBSET] [-P POLL_INTERVAL] [-B SECONDS] [-o] [-
t TREE] [--private-key PRIVATE_KEY_FILE] [-u REMOTE_USER] [-c CONNECTION] [-T
 TIMEOUT]
                           [--ssh-common-args SSH_COMMON_ARGS] [--sftp-extra-
args SFTP_EXTRA_ARGS] [--scp-extra-args SCP_EXTRA_ARGS] [--ssh-extra-args
 SSH_EXTRA_ARGS] [-k |
                           --connection-password-file CONNECTION_PASSWORD_FILE]
 [-C] [-D] [-e EXTRA_VARS] [--vault-id VAULT_IDS] [-J | --vault-password-file
 VAULT_PASSWORD_FILES] [-f FORKS]
                           [-M MODULE_PATH] [--playbook-dir BASEDIR] [--task-timeout
 TASK_TIMEOUT] [-a MODULE_ARGS] [-m MODULE_NAME]
                          pattern
                          Define and run a single task 'playbook' against a set of
 hosts
                          [...]
```

`ansible-config`

Used to manage and view Ansible's configuration settings.

```
> ansible-config -h
            usage: ansible-config [-h] [--version] [-v]
 {list,dump,view,init,validate} ...

View ansible configuration.

positional arguments:
  {list,dump,view,init,validate}
    list                Print all config options
    dump                Dump configuration
    view                View configuration file
    init                Create initial configuration
    validate            Validate the configuration file and environment variables.
 By default it only checks the base settings without accounting for plugins (see -
t).
```

```
options:
  --version             show program's version number, config file location,
 configured module search path, module location, executable location and exit
  -h, --help            show this help message and exit
  -v, --verbose         Causes Ansible to print more debug messages. Adding multiple
 -v will increase the verbosity, the builtin plugins currently evaluate up to -
vvvvvv. A
                        reasonable level to start is -vvv, connection debugging
 might require -vvvv. This argument may be specified multiple times.
```

ansible-console

An interactive command-line tool or Read-Eval-Print Loop (REPL) that lets you run Ansible ad hoc commands in a persistent real-time shell environment. It is a great tool for quickly testing modules, exploring your inventory and debugging issues without having to run a new command for every action.

```
> ansible-console -h
            usage: ansible-console [-h] [--version] [-v] [-b] [--become-method
 BECOME_METHOD] [--become-user BECOME_USER] [-K | --become-password-file
 BECOME_PASSWORD_FILE] [-i INVENTORY]
            [--list-hosts] [-l SUBSET] [--private-key PRIVATE_KEY_FILE] [-u
 REMOTE_USER] [-c CONNECTION] [-T TIMEOUT] [--ssh-common-args SSH_COMMON_ARGS]
            [--sftp-extra-args SFTP_EXTRA_ARGS] [--scp-extra-args SCP_EXTRA_ARGS]
 [--ssh-extra-args SSH_EXTRA_ARGS] [-k |
            --connection-password-file CONNECTION_PASSWORD_FILE] [-C] [-D] [--vault-
id VAULT_IDS] [-J | --vault-password-file VAULT_PASSWORD_FILES] [-f FORKS]
            [-M MODULE_PATH] [--playbook-dir BASEDIR] [-e EXTRA_VARS] [--task-
timeout TASK_TIMEOUT] [--step]
            [pattern]

REPL console for executing Ansible tasks.
                          [...]
```

ansible-doc

A command-line tool that provides documentation for all Ansible's installed modules and plug-ins. It is an essential utility for anyone writing playbooks or ad hoc commands. You can look up a module's purpose, parameters, return , and even usage examples directly from your terminal.

```
> ansible-doc -h
            usage: ansible-doc [-h] [--version] [-v] [-M MODULE_PATH] [--playbook-
dir BASEDIR]
                   [-t
 {become,cache,callback,cliconf,connection,httpapi,inventory,lookup,netconf,shell,vars,module,stra
 [-j] [-r ROLES_PATH]
```

```
                    [-e ENTRY_POINT | -s | -F | -l | --metadata-dump] [--no-fail-on-
errors]
                    [plugin ...]

plugin documentation tool
                           [...]
```

`ansible-galaxy`

A command-line tool that comes with Ansible and is used for managing and sharing roles.

```
> ansible-galaxy -h
            Perform various Role and Collection related operations.

positional arguments:
  TYPE
    collection   Manage an Ansible Galaxy collection.
    role         Manage an Ansible Galaxy role.

options:
  --version      show program's version number, config file location, configured
 module search path, module location, executable location and exit
  -h, --help     show this help message and exit
  -v, --verbose  Causes Ansible to print more debug messages. Adding multiple -v
 will increase the verbosity, the builtin plugins currently evaluate up to -vvvvvv.
 A reasonable level
                 to start is -vvv, connection debugging might require -vvvv. This
 argument may be specified multiple times.
```

`ansible-inventory`

Is a command-line tool that helps you inspect, validate and understand your inventory.

```
> ansible-inventory -h
            usage: ansible-inventory [-h] [--version] [-v] [-i INVENTORY] [-l
 SUBSET] [--vault-id VAULT_IDS] [-J | --vault-password-file VAULT_PASSWORD_FILES]
 [--playbook-dir BASEDIR]
                          [-e EXTRA_VARS] [--list] [--host HOST] [--graph] [-y] [--
toml] [--vars] [--export] [--output OUTPUT_FILE]
                          [group]

Show Ansible inventory information, by default it uses the inventory script JSON
 format
[...]
```

`ansible-playbook`

Is the primary command-line tool used to run Ansible's automation blueprints, also known as playbooks, against an inventory of target hosts.

```
> ansible-playbook -h
            usage: ansible-playbook [-h] [--version] [-v] [--private-key
 PRIVATE_KEY_FILE] [-u REMOTE_USER] [-c CONNECTION] [-T TIMEOUT] [--ssh-common-args
 SSH_COMMON_ARGS]
                         [--sftp-extra-args SFTP_EXTRA_ARGS] [--scp-extra-args
 SCP_EXTRA_ARGS] [--ssh-extra-args SSH_EXTRA_ARGS] [-k |
                         --connection-password-file CONNECTION_PASSWORD_FILE] [--
force-handlers] [--flush-cache] [-b] [--become-method BECOME_METHOD] [--become-user
 BECOME_USER] [-K |
                         --become-password-file BECOME_PASSWORD_FILE] [-t TAGS]
 [--skip-tags SKIP_TAGS] [-C] [-D] [-i INVENTORY] [--list-hosts] [-l SUBSET] [-e
 EXTRA_VARS]
                         [--vault-id VAULT_IDS] [-J | --vault-password-file
 VAULT_PASSWORD_FILES] [-f FORKS] [-M MODULE_PATH] [--syntax-check] [--list-tasks]
 [--list-tags] [--step]
                         [--start-at-task START_AT_TASK]
                         playbook [playbook ...]

Runs Ansible playbooks, executing the defined tasks on the targeted hosts.
[...]
```

`ansible-vault`

Is a command-line tool that allows you to encrypt sensitive data, such as passwords, API keys and other secrets and store them securely within your playbooks and other configuration files.

```
> ansible-vault -h
            usage: ansible-vault [-h] [--version] [-v]
 {create,decrypt,edit,view,encrypt,encrypt_string,rekey} ...

encryption/decryption utility for Ansible data files

[...]
```

5.3 Configuring SSH

By default, Ansible assumes SSH keys are being used between a control and managed nodes to establish a connection. To set up SSH key-based authentication from the control node to the managed nodes:

1. Generate a key pair on the control node.
   It is recommended to use a dedicated SSH key that is clearly defined in the variable `ansible_ssh_private_key_file`.
2. Copy the public key to the managed node.

```
> ssh-copy-id -i PATH_TO_PUBLIC_KEY TARGET_USER@IP_ADDRESS/HOSTNAME
```

This onetime setup allows Ansible to connect securely without prompting for credentials during automation.

3. Add the private key to the control node.

```
> ssh-add PATH_TO_PRIVATE_KEY
```

5.4 Creating a playbook

An Ansible playbook is a `YAML` file that defines a series of tasks to be executed on one or more hosts. This topic covers the creation of a simple playbook.

1. Define an inventory file, for example:

```
[all]
192.168.122.154 ansible_user=user1
```

Although Ansible uses `/etc/ansible/hosts` by default. It is a best practice to maintain a separate inventory file and specify it using the `--inventory` flag.

2. Verify your inventory file, for example:

```
> ansible-inventory --inventory test.ini --list
```

3. Test the connection between the control node and managed nodes with the Ansible ad hoc command:

```
> ansible all -i test.ini -m ping
```

If the configuration is correct, the following message is displayed:

```
        [WARNING]: Platform linux on host 192.168.122.73 is using the discovered
 Python interpreter at /usr/bin/python3.13, but future installation of another
 Python interpreter could change
        the meaning of that path. See https://docs.ansible.com/ansible-core/2.18/
reference_appendices/interpreter_discovery.html for more information.
        192.168.122.73 | SUCCESS => {
            "ansible_facts": {
                "discovered_interpreter_python": "/usr/bin/python3.13"
            },
```

This confirms Ansible can reach the managed nodes securely via SSH.

4. Playbooks are `YAML` files that define the tasks Ansible should perform. For example, create a file named `install.yml` to define a task to install common packages on all managed SUSE systems:

```
- name: Ensure essential packages are present
  hosts: all
  become: true

  tasks:
    - name: Install vim and curl
      ansible.builtin.package:
        name:
          - suseconnect
          - curl
        state: present
```

5. Run the playbook in check mode:

```
> ansible-playbook install.yml --check
```

Check mode is a simulation. It does not generate output for tasks. It validates a configuration management playbook and is useful for simple playbooks.

6. Run the playbook:

```
> ansible-playbook -i test.ini install.yml
```

Ansible connects to each host in your inventory and executes the defined tasks using the system's package manager.

5.5 Understanding roles

Roles allow you to easily reuse and share your Ansible Automation by using a known file structure to automatically load all related artifacts, including tasks, variables, files, and handlers.

5.5.1 Using roles

You can use the role in the following ways:

- At the play level with the `roles` option: This is the default way of using roles in a play.
- At the task level with `ansible.builtin.include_role`: You can reuse roles dynamically anywhere in the `tasks` section of a play using `ansible.builtin.include_role`.

- At the task level with `ansible.builtin.import_role`: You can reuse roles statically anywhere in the `tasks` section of a play using `ansible.builtin.import_role`.
- As a dependency of another role.

A play is the fundamental execution unit within an Ansible playbook.

5.5.2 Importing roles

You can reuse roles statically anywhere in the `tasks` section of a play by using `import_role`. For example:

```
---
- hosts: webservers
  tasks:
    - name: Print a message before executing role
      ansible.builtin.debug:
        msg: "Debug message before executing role"

    - name: Apply standard user and group setup from 'base_users' role
      ansible.builtin.import_role:
        name: base_users

    - name: Print a message after executing role
      ansible.builtin.debug:
        msg: "Debug message after executing role"
```

You can use other keywords, for example:

```
---
- hosts: webservers
  tasks:
    - name: Import the foo_app_instance role
      ansible.builtin.import_role:
        name: test_app_instance
      vars:
        dir: '/opt/a'
        app_port: 5000
```

5.6 About variables

With Ansible, you can run tasks and playbooks with a single command. Variables are placeholders for values that allow you to make your playbooks dynamic, reusable, and flexible. For example, instead of hard-coding values like a username, port number, or a list of packages, directly into your tasks, you can use a variable. You can define these variables using standard YAML syntax (including lists and dictionaries) in your playbooks, inventories, roles, or at the command line. You can also create new variables dynamically during a playbook run.

5.6.1 Variable placement

When you define the same variable in multiple locations, Ansible uses variable precedence to determine which value to apply. This fixed order governs how different variable sources override one another during execution. Variable precedence is defined in the following order, from least to greatest (the last listed variable overrides all other variables):

- Command-line values
- Role defaults
- Inventory file or script group vars
- Inventory `group_vars/all`
- Playbook `group_vars/all`
- Inventory `group_vars/*`
- Playbook `group_vars/*`
- Inventory file or script host vars
- Inventory `host_vars/*`
- Playbook `host_vars/*`
- Host facts and cached `set_facts`
- Play `vars`
- Play `vars_prompt`
- Play `vars_files`

- Role vars
- Block vars
- Task vars
- `include_vars`
- Registered vars and `set_facts`
- Role and `include_role` params
- include params
- extra vars

Ansible gives precedence to variables defined more recently, more active and with more explicit scope. Variables inside the `default` folder are easily overridden. Anything in the `vars` directory of the role overrides previous versions of that variable in the namespace. For more on variable precedence, refer to https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_variables.html#variable-precedence-where-should-i-put-a-variable.

5.6.2 Setting variables

You can define variables in an inventory, in playbooks, in reusable files, in roles and at the command line. Ansible loads all the variables it finds and then chooses which variable to apply based on variable precedence.

- **Defining variables in an inventory**: You can either define different variables for each host or set shared variables for a group of hosts in your inventory. For example, if all systems in the `manhattan` group use `manhattan.ntp.example.com` as an NTP server, you can use a group variable.
- **Defining variables in a play**: You can define variables directly in a play. When you define variables directly, they are visible only to the tasks executed in that play.
- **Defining variables in include files and roles**: You can define variables in reusable variable files or reusable roles. If you define variables in reusable variable files or roles, the sensitive variables are separated from playbooks. This separation enables you to store your playbooks in a source control software and even share the playbooks, without the risk of exposing passwords or other sensitive and personal data. For example:

```
---
```

```
- hosts: all
  remote_user: root
  vars:
    favcolor: blue
  vars_files:
    - /vars/test.yml

  tasks:

  - name: This is just a placeholder
    ansible.builtin.command: /bin/echo foo
```

The content of each variable file is a simple YAML dictionary. For example, `/vars/test.yml` :

```
- name: Show foo and bar
  ansible.builtin.debug:
    msg:
      foo: "{{ foo }}"
      bar: "{{ bar }}"
```

6 Ansible best practices

To get the most out of Ansible and Ansible playbooks , here are some best practices you can follow:

- **Group by roles**: A system can belong to multiple groups, a powerful concept exemplified by naming groups descriptively, such as `webservers` and `dbservers`. This structure enables playbooks to target machines based on their role and to assign role-specific variables through the group variable system.
- **Describe the tasks**: It is recommended to provide a clear, descriptive `name` for the `tasks`. This description explains why the task is being performed and is displayed when the playbook executes.
- **Use the `name` directive**: It is a good practice to use the `name` directive to name your plays, tasks and blocks.
- **Simplicity**: Keep it simple and use only the Ansible features you need.

6.1 Protect sensitive data with the `ansible-vault` command-line tool

With Vault, you can encrypt variables and files so you can protect sensitive content such as passwords or keys rather than leaving it visible as plaintext in playbooks or roles. Some best practices include:

- **Managing Vault passwords**: To keep track and manage your Vault , it is best to have a strategy. For example, for a small project or a few sensitive values, you can use a single password for encryption. You can use multiple passwords for a large project or many sensitive values. You can distinguish between different passwords with Vault IDs. Based on your use case, you might want a different password for each encrypted file, for each directory, or for each environment. There are two options within Ansible, to store Vault passwords: in files or using a third-party tool, such as a system keyring or a secret manager. For passwords stored in a third-party tool, you need a vault password client script to retrieve them from within Ansible.

- **Encrypting individual variables or files**: The two types of encryption content in Ansible are variables and files. Encrypted content is identified with the `!vault` tag, which tells Ansible and YAML that the content needs to be decrypted. The `|` character is used for multi-line strings. Ansible Vault can encrypt any structured data file used by Ansible and the encryption is done in the Vault.

- **Using encrypted individual variables or files**: You must provide passwords for encrypted variables or files when running a task or playbook that uses them. You can supply these passwords either directly at the command line or by configuring a default password source using a configuration option or an environment variable. For a single password , use `--ask-vault-pass` or `--vault-password-file` options. When you use multiple vaults in a single inventory, use the `--vault-id option.` You can also use multiple `--vault-id` options to specify the Vault IDs.

7 Ansible support coverage

The support matrix for Ansible and Python compatibility:

- **Control node:**
  - Python 3.11 - 3.13
  - ansible-core 2.18
  - Ansible 11
- **Managed node:**
  - Python 3.11 - 3.13

Important

- Python and Ansible versions in the control and managed nodes must be compatible.
- Third-party software is not supported.
- Only versions provided in packages that install into System Python are supported.
- Python Virtual Environment or alternatives like pyenv are not supported.

8 More information

To learn more about Ansible, refer to The official source for Ansible core. (https://docs.ansible.com/ansible-core/devel/index.html)

9 Legal Notice

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither SUSE LLC, its affiliates, the authors, nor the translators shall be held liable for possible errors or the consequences thereof.

A GNU Free Documentation License

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download

using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See https://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been

published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

```
Copyright (c) YEAR YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled “GNU
Free Documentation License”.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with...Texts.” line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.