SUSE

Ansible Linux System Roles

WHAT?

This article gives an introduction to various Ansible Linux system roles that help to automate the configuration and management on SUSE Linux Enterprise Server 16.0 systems.

WHY?

Learn how to automate IT infrastructure with Ansible Linux system roles.

EFFORT

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

REQUIREMENTS

- *Linux fundamentals:* Understanding basic Linux commands,file permissions, directory structures and usage 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 Nov 2025

Contents

1 About Ansible Linux system roles

Linux system roles are a set of Ansible roles designed to automate and configure common components and services of the Linux operating system.

Linux system roles are always used with Ansible playbooks. You define the desired state of your systems in an Ansible playbook, specifying which roles to apply and with what parameters. Ansible then connects to your Linux hosts and executes the tasks defined within the roles to bring your systems into the desired state.

The system roles are shipped in the `ansible-linux-system-roles` package on SUSE Linux Enterprise Server 16.0 systems.

Once the `ansible-linux-system-roles` package is installed, you can access:

- *Roles*:`/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/roles`
- *Documentation*:`/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs`

Each role has a `README` in the `docs` directory that includes:

- Description of the role
- Supported variables and their usage
- Example playbooks

The `README.md` files for each role:

```
/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs # ls
    CHANGELOG_aide.md            CHANGELOG_timesync.md     README_suseconnect.md
    CHANGELOG_certificate.md     README_aide.md            README_systemd.md
    CHANGELOG_cockpit.md         README_certificate.md     README_timesync.md
    CHANGELOG_crypto_policies.md README_cockpit.md         aide
    CHANGELOG_firewall.md        README_crypto_policies.md certificate
    CHANGELOG_ha_cluster.md      README_firewall.md        firewall
    CHANGELOG_journald.md        README_ha_cluster.md      ha_cluster
    CHANGELOG_keylime_server.md  README_journald.md        journald
    CHANGELOG_mssql.md           README_keylime_server.md  keylime_server
    CHANGELOG_podman.md          README_mssql.md           podman
    CHANGELOG_postfix.md         README_podman.md          selinux
    CHANGELOG_selinux.md         README_postfix.md         ssh
    CHANGELOG_ssh.md             README_selinux.md         systemd
```

```
CHANGELOG_systemd.md        README_ssh.md            timesync
```

2 Installing Ansible Linux system roles

To install Ansible Linux system roles on a SUSE Linux Enterprise Server control node:

- Install the roles on a control node:

```
> sudo  zypper install ansible-linux-system-roles
```

The control node is where Ansible and the Linux system roles are installed. It is not required to have Ansible or Linux system roles installed on managed nodes.

3 About the firewall Linux system role

The firewall Linux system role in Ansible is a pre-packaged and reusable set of tasks and defaults designed to configure and manage the `firewalld` service on SUSE Linux Enterprise Server 16.0 systems.

You can use this role to:

- Install and enable the `firewalld` service.
- Manage zones such as `public`, `home` or `internal.`
- Add or remove services such as `ssh` to specific zones.
- Open or close ports, for example, `8080` or `TCP` for particular zones.
- Configure port forwarding and masquerading (NAT).
- Handle permanent and runtime configuration.

EXAMPLE 1: RESET FIREWALL AND ENABLE SSH SERVICE

```
---
- name: Erase existing config and enable ssh service
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
```

```
    firewall:
      - previous: replaced
      - service: ssh
        state: enabled

  tasks:
    - name: reset firewall and enable SSH service
      ansible.builtin.include_role:
        name: suse.linux_system_roles.firewall
```

- `previous: replaced`: Resets the `firewalld` configuration by removing all existing user-defined settings. `previous` is a `string` datatype and accepts value `replaced`.
- `service: ssh`: Secure Shell (SSH) as the specific network traffic type to be managed. `service` is a `string` datatype and accepts values `ssh`,`http`,`tftp`etc.
- `state: enabled`: Permanently opens the firewall for incoming SSH traffic. `state` is a `string` datatype and accepts values `enabled` or `disabled`.

EXAMPLE 2: DISABLE TFTP SERVICE

```
---
- name: Disable TFTP service on managed nodes
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
    firewall:
      - service: tftp
        state: disabled
  tasks:
    - name: Configure firewall to disable TFTP service
      ansible.builtin.include_role:
        name: suse.linux_system_roles.firewall
```

- `service: tftp`: Defines the Trivial File Transfer Protocol (TFTP) as the specific network service to be managed. `service` is a `string` datatype and accepts values `ssh`,`httptftp`etc.
- `state: disabled`: Permanently block all incoming network traffic related to `tftp`. `state` is a `string` datatype and accepts values `enabled` or `disabled`

For more details about all the variables in the firewall Linux system role, refer to the specific `README.md` file on the control node:

```
/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs
```

4 About the AIDE Linux system role

The AIDE Linux system role in Ansible installs, configures and manages the Advanced Intrusion Detection Environment (AIDE) utility on SUSE Linux Enterprise Server 16.0 systems.

You can use this role to:

- Ensure the AIDE package is installed on the managed node.
- Optionally generate and deploys a custom `/etc/aide.conf`file.
- Create the baseline AIDE database on the managed node.
- Run checks against the live system and reports any file additions, deletions, or modifications. This is used for daily intrusion detection.
- Securely fetch the AIDE database files from the remote host back to the control node.

EXAMPLE 3: AIDE ROLE

```
---
- name: Example aide role invocation
  hosts: managed_hosts
  become: true
  gather_facts: true

  tasks:
    - name: Include role aide
      vars:
        aide_db_fetch_dir: files
        aide_init: true
        aide_check: false
        aide_update: false
      ansible.builtin.include_role:
        name: suse.linux_system_roles.aide
```

- `aide_db_fetch_dir: files`: is a `string` datatype that specifies the directory on control node, where the AIDE database files are fetched and stored from the remote managed nodes. The default is `files`, which means the role expects a directory named `files` to exist in the same directory as the playbook being executed.
- `aide_init: true`: is a `Boolean` datatype that controls whether a AIDE database should be initialized on a managed node. The default is `false`. When set to `true`, a AIDE database is created on the managed node, it is then fetched to the control node.

- `aide_check: false`: is a `Boolean` datatype that controls whether an integrity check should be immediately run on the managed nodes. The default is `false`. When set to `true`, the AIDE check is run, data is compared with the current state of the file system with the trusted baseline database and a report is generated.
- `aide_update: false`: is a `Boolean` datatype that controls the creation of a new AIDE database to replace the current trusted baseline. The default is `false`. When set to `true`, the AIDE update command is run on the managed system, which performs an integrity check, creates a new AIDE database reflecting the current system state and establishes this new file as the trusted baseline for future checks.

For more details about all the variables in the AIDE Linux system role, refer to the specific `README.md` file on the control node:

```
/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs
```

5 About the Cockpit Linux system role

The Cockpit Linux system role in Ansible automates the installation and configuration of the Cockpit Web Console, a web-based graphical interface for administering SUSE Linux Enterprise Server 16.0 systems.

You can use this role to:

- Ensure the necessary Cockpit packages are installed.
- Ensure `cockpit.socket` is enabled for socket-based activation.
- Allow the administrator to set specific options in the `/etc/cockpit/cockpit.conf`file, such as the login title or session timeouts `cockpit_config`.

EXAMPLE 4: INSTALL COCKPIT ON ALL HOSTS

```
---
- name: install Cockpit on all hosts
  hosts: managed_nodes
  become : true
  gather_facts: true

  tasks:
```

```
    - name: Dynamically execute the Cockpit role
      ansible.builtin.include_role:
        name: suse.linux_system_roles.cockpit
```

EXAMPLE 5: INSTALL COCKPIT ON ALL HOSTS AND AUTOMATICALLY OPEN THE FIREWALL PORT

```
---
- name: Install Cockpit and automatically open firewall port
  hosts: managed_hosts
  become: true
  gather_facts: true

  vars:
    cockpit_enabled: true
    cockpit_started: true
    cockpit_manage_firewall: true

  tasks:
    - name: Dynamically install Cockpit and open firewall
      ansible.builtin.include_role:
        name: suse.linux_system_roles.cockpit
```

- `cockpit_enabled: true`: controls the `systemd` service setting for the Cockpit Web Console. `cockpit_enabled` is a `Boolean` datatype and accepts values `true` and `false`. Default is `true`, which means the Cockpit service is configured to start automatically at system boot.
- `cockpit_started: true`: controls the immediate runtime status of the Cockpit Web Console service. `cockpit_started` is a `Boolean` datatype and accepts values `true` and `false`. Default is `true`, which means the Cockpit service is running immediately on the managed host.
- `cockpit_manage_firewall: true`: delegates the task of configuring the firewall to the role itself, ensuring the Cockpit web interface is network-accessible. `cockpit_manage_firewall` is a `Boolean` datatype and accepts values `true` and `false`. Default is `false`, which means the role installs and enables Cockpit but makes no changes to the firewall, leaving that task to be handled manually or by a separate dedicated firewall role.

For more details about all the variables in the Cockpit Linux system role, refer to the specific `REDADME.md` file on the control node:

```
/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs
```

6 About the HA cluster Linux system role

The HA cluster Linux system role in Ansible is designed to fully install, configure and manage a Pacemaker and Corosync High Availability (HA) cluster on SUSE Linux Enterprise Server 16.0 systems.

You can use this role to:

- Automate the installation of core cluster components like Pacemaker and Corosync.
- Create and initialize a robust, multi-node High Availability cluster.
- Configure fencing devices to ensure true node isolation and data integrity upon failure.
- Define and manage cluster resources, such as floating IP addresses, application services, and shared storage mounts.
- Set resource constraints and dependencies to control which nodes run specific services.
- Enable and manage cluster services across the defined cluster nodes.
- Ensure consistent HA setup across multiple environments, adhering to best practices and minimizing manual errors.

EXAMPLE 6: CONFIGURE FIREWALL AND SELINUX FOR MANAGED NODES

```
---
- name: Configure firewall and selinux for managed nodes
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
    ha_cluster_manage_firewall: true
    ha_cluster_manage_selinux: true

  tasks:
    - name: Manage firewall and selinux
      ansible.builtin.include_role:
        name: suse.linux_system_roles.ha_cluster
```

- `ha_cluster_manage_firewall: true`: It controls whether the role should automatically configure the system's firewall to permit necessary cluster communication. `ha_cluster_manage_firewall` is a `Boolean` datatype and accepts values `true`,`false`,`yes`,`no`, `0` and `1`. When set to `true`, it automatically adds firewall rules to open the necessary ports for the cluster to function. Default is `false`.
- `ha_cluster_manage_selinux: true`: Controls whether the HA cluster role will automatically configure the SELinux policy to allow cluster processes to function. `ha_cluster_manage_selinux` is a `Boolean` datatype and accepts values `true`,`false`,`yes`,`no`, `0` and `1`. When set to `true`, it manages SELinux and performs actions like setting the correct SELinux contexts on necessary files and directories, enabling or setting SELinux booleans required for cluster services etc. Default is `false`.

EXAMPLE 7: CREATE PCSD TLS CERTIFICATES AND KEY FILES

```
---
- name: Manage HA cluster with TLS certificates and key files
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
    ha_cluster_pcsd_certificates:
      - name: FILENAME
        common_name: "{{ ansible_hostname }}"
        ca: self-sign

  tasks:
    - name: Manage firewall and selinux
      ansible.builtin.include_role:
        name: suse.linux_system_roles.ha_cluster
```

This example creates self-signed `pcsd` certificate and private key files in `/var/lib/pcsd` with the file name `FILENAME.crt` and `FILENAME.key`.

- `ha_cluster_pcsd_certificates`: is a list of dictionaries used to define TLS certificates for the pcsd service. The default is `[]`. It uses the structure of the underlying `certificate` system role. The certificate specific keys are `ca`, `common_name` and `cert_basename`. These keys are all `string` data types.

For more details about all the variables in the HA cluster Linux system role, refer to the specific `README.md` file on the control node:

```
/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs
```

7 About the SUSEconnect Linux system role

The SUSEconnect Linux system role in Ansible manages SUSE Linux system registrations with SUSE Customer Center or a SMT server. It automates the process of registering and deregistering systems, as well as managing additional products and modules on a SUSE system.

You can use this role to:

- Register a SUSE system to the SCC or servers.
- Activate or remove of specific add-on products or modules.
- Deregister systems or products.
- Support transactional update register SLE-MICRO.
- Recheck tasks to ensure a smooth registration process.
- Ensure compatibility with public cloud environments.

EXAMPLE 8: REGISTER A SUSE LINUX ENTERPRISE SERVER 16.0 SYSTEM

```
---
- name: Register with SCC and activate modules
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
    suseconnect_base_product:
      key: '{{ sles_registration_key }}'
      product: '{{ ansible_distribution }}'

    suseconnect_subscriptions:
      - {name: "sle-module-containers", state: enabled}
      - {name: "PackageHub", state: disabled}
      - {name: "sle-module-python3", state: enabled}

  tasks:
    - name: Register system and modules with SUSE Customer Center
      ansible.builtin.include_role:
        name: suse.linux_system_roles.suseconnect
```

- `suseconnect_base_product`: defines the required parameters for registering the core operating system of a SUSE Linux machine. `suseconnect_base_product` is defined as a YAML Dictionary containing key-value pairs that map directly to the requirements of SCC or SMT servers. Accepted string values are `key`, `product`, `version` and `arch`.

- `suseconnect_subscriptions`: is a crucial component for enterprise SUSE deployments, as it controls which features, for example, High Availability, containers, or web application platforms enabled on the registered base operating system. `suseconnect_subscriptions` is defined as a YAML Dictionary containing key-value pairs that map directly to the requirements of SCC or SMT servers. Accepted string values are `name`, `state`, `version` and `key`.

EXAMPLE 9: DEREGISTER BASE PRODUCTS

```
---
- name: Deregister products from SCC
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
    suseconnect_deregister: true

  tasks:
    - name: Deregister products from SUSE Customer Center
      ansible.builtin.include_role:
        name: suse.linux_system_roles.suseconnect
```

- `suseconnect_deregister: true`: instructs the system to deactivate its base subscription and remove it from the the SCC or SMT servers. `suseconnect_deregister` is a `Boolean` datatype and accepts values `false` and `true`. When set to `true`, it executes the necessary SUSEConnect command to deregister the system's base product subscription. Default is `false`.

For more details about all the variables in the SUSEConnect Linux system role, refer to the specific `README.md` file on the control node:

```
/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs
```

8 About the Journald Linux system role

The Journald Linux system role in Ansible is designed to configure and manage the `systemd-journald` service on SUSE Linux Enterprise Server 16.0 systems.

The role automates changes to `/etc/systemd/journald.conf`file. You can use this role to:

- Determine where journal logs are stored. Logs can be stored temporarily in memory or persistently on the disk.
- Set hard limits on how much disk space the journal can consume before old logs are deleted.
- Configure how much disk space the journal should always leave free for other system uses.
- Control whether journal data objects should be compressed to save space.
- Configure whether log data is kept separate for individual users.
- Manage the running state of the `systemd-journald`service.

EXAMPLE 10: CONFIGURE PERSISTENT STORAGE, LIMIT DISK USAGE, ENABLE USER LOGGING AND SET SYNCHRONIZATION FREQUENCY

```
 ---
 - name: Configure persistent storage, limit disk usage, enable user logging and
set synchronization frequency
   hosts: managed_nodes
   become: true
   gather_facts: true

   vars:
        journald_persistent: true
        journald_max_disk_size: 1024
        journald_per_user: true
        journald_sync_interval: 1

   tasks:
     - name: Configure persistent storage, limit disk usage, enable user logging
and set synchronization frequency
       ansible.builtin.include_role:
         name: suse.linux_system_roles.journald
```

- `journald_persistent: true`: instructs journald to create the `/var/log/journal` directory and store logs permanently. `journald_persistent` is a `Boolean` datatype and accepts values accepts values `true` and`false`. Default is `false`.
- `journald_max_disk_size: 1024`: limits the total disk space the journal files may use up to 1 Gigabyte (1024 MB). `journald_max_disk_size` is an `integer` datatype and accepts an integer representing a size in megabytes (MB). No value is explicitly configured by this role; therefore, the default sizing logic from `man 5 journald.conf` applies.
- `journald_per_user: true`: Enables per-user logging. Logs are kept separate for individual users, allowing unprivileged users to read logs related only to their own services. `journald_per_user` is a `Boolean` datatype and accepts values `true`,`false`,`yes`,`no`, `0` and `1`. Default is `false`.
- `journald_sync_interval: 1`: limits the total disk space the journal files may use up to 1 Gigabyte (1024 MB). `journald_max_disk_size` is an `integer` datatype and accepts an integer representing a time span in minutes. By default role does not alter currently used value.

EXAMPLE 11: CONFIGURE JOURNALD FOR PERSISTENT STORAGE AND DISK LIMITS

```
---
   - name: Configure systemd-journald for persistent storage and disk limits
     hosts: managed_nodes
     become: true
     gather_facts: true

     vars:
       journald_persistent: true
       journald_max_disk_size: 1024

     tasks:
       - name: Configure journald for persistent storage and disk limits
         ansible.builtin.include_role:
           name: suse.linux_system_roles.journald
```

- `journald_persistent: true`: instructs journald to create the `/var/log/journal` directory and store logs permanently. `service` is a `Boolean` datatype and accepts values accepts values `true`,`false`,`yes`,`no`, `0` and `1`. Default is `false`.
- `journald_max_disk_size: 1024`: limits the total disk space the journal files may use up to 1 Gigabyte (1024 MB). `journald_max_disk_size` is an `integer` datatype and accepts an integer representing a size in megabytes (MB). No value is explicitly configured by this role; therefore, the default sizing logic from `man 5 journald.conf` applies.

For more details about all the variables in the journald Linux system role, refer to the specific `README.md` file on the control node:

```
/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs
```

9 About the Podman Linux system role

The Podman Linux system role in Ansible automates the deployment and lifecycle management of containers as system services at scale, ensuring consistent and often rootless container orchestration across an infrastructure.

You can use this role to:

- Ensure the necessary Podman packages and related tools are installed and updated.
- Automate the prerequisite setup for running containers as unprivileged users. Also includes managing the correct entries in `/etc/subuid` and `/etc/subgid`files.
- Manage the creation, starting, stopping and removal of containers.
- Leverage Podman's deep integration with `systemd` by automatically generating `systemd` unit files (or Quadlet files on newer systems) to manage containers and pods as persistent services that start automatically at boot.
- Deploy complex multi-container workloads defined using standard Kubernetes YAML manifests.

EXAMPLE 12: CREATE A CONTAINER RUNNING AS ROOT WITH PODMAN VOLUME

```
---
- name: Manage podman root containers and services
  hosts: managed_hosts
  become: true
  gather_facts: true

  vars:
    podman_firewall:
      - port: 8080/tcp
        state: enabled
    podman_kube_specs:
      - state: started
        kube_file_content:
          apiVersion: v1
```

```
          kind: Pod
          metadata:
            name: test1-httpd
          spec:
            containers:
              - name: test1-httpd
                image: registry.access.suse.com/test1/httpd-24
                ports:
                  - containerPort: 8080
                    hostPort: 8080
                volumeMounts:
                  - mountPath: /var/www/html:Z
                    name: test1-html
            volumes:
              - name: test1-html
                persistentVolumeClaim:
                  claimName: test1-html-volume

  tasks:
    - name: Dynamically install Cockpit and open firewall
      ansible.builtin.include_role:
        name: suse.linux_system_roles.podman
```

In this example , a Kubernetes pod `test1-httpd` running an HTTP server container from the `registry.access.suse.com/test1/httpd-24` image. The container's web content is mounted from a persistent volume named `test1-html-volume`. By default, the role creates rootful containers.

- `podman_firewall`: controls the necessary firewall configurations on the host to ensure container ports are accessible. It is a list of dictionaries, where each dictionary defines a single firewall rule for a container. The accepted values are specified as key-value pairs within these dictionaries. The most common and crucial keys accepted within each dictionary item are `port`, `state`, `zone` and `masquerade`. Default is an empty list `[]` or `null`.
- `podman_kube_specs`: defines and manage Podman containers or pods based on Kubernetes YAML specifications. It is a list of dictionaries and each dictionary item defines one container or pod deployment and requires one of the following keys to specify the Kubernetes YAML source: `kube_file_src` or `kube_file_content`. Default is an empty list `[]` or `null`.

For more details about all the variables in the Podman Linux system role, refer to the specific `README.md` file on the control node:

```
/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs
```

10 About the certificate Linux system role

The certificate Linux system role in Ansible automates the entire lifecycle of TLS and SSL certificates on managed Linux hosts.

You can use this role to:

- Generate the private key and the Certificate Signing Request (CSR) locally on the managed node, ensuring the private key never leaves the host.
- Handle the submission of the CSR to a designated Certificate Authority (CA).
- Save the newly issued certificate, key, and CA chain files to the correct, secure locations on the file system, managing file ownership and permissions.
- Set up mechanisms such as `certmonger` to automatically monitor the certificate's expiration date and attempt renewal before it expires.

EXAMPLE 13: ISSUE A SELF SIGNED CERTIFICATE

```
---
-name: Issue a self signed certificate
 hosts: managed_nodes
 become: true

  vars:
    certificate_requests:
      - name: mycert
        dns: *.example.com
        ca: self-sign

  tasks:
    - name: Issue a self signed certificate
      ansible.builtin.include_role:
        name: suse.linux_system_roles.certificate
```

- `certificate_requests`: defines the specific details of the certificates the role needs to issue, manage, or renew. It is a list of dictionaries and each item in the list is a dictionary specifies the properties of one unique certificate to be managed. The default is an empty list `[]`. If this list is empty, the role performs no certificate management actions. Each dictionary within the `certificate_requests` list must define at least the certificate name and the

issuing Certificate Authority. Other parameters specify the key size, subject, and domains. In this example, `name`, `ca` are `string` data types and required. `dns` can be either a `list` or `string` and is optional.

You can find the directory at:

- Certificates: `/etc/ssl/certs`
- Keys: `/etc/ssl/certs/private`

EXAMPLE 14: ISSUE A CERTIFICATE AND KEY AND SPECIFY LOCATION TO PLACE THEM

```
---
-name: Issue a certificate and specify location
 hosts: managed_nodes
 become: true
 gather_facts: true

 vars:
   certificate_requests:
     - name: test/path/mycert
       dns: *.example.com
       ca: self-sign

  tasks:
    - name: Issue a certificate and specify location
      ansible.builtin.include_role:
        name: suse.linux_system_roles.certificate
```

This example creates a certificate file in `/test/path/mycert.crt` and a key file in `/test/path/mycert.key` .

For more details about all the variables in the certificate Linux system role, refer to the specific `README.md` file on the control node:

```
/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs
```

11 About the Crypto policies Linux system role

The Crypto policies system role in Ansible is to establish the rules, guidelines, and standards for the proper and effective use of cryptography on SUSE Linux Enterprise Server 16.0 systems.

You can use this role to:

- Specify when and how different types of data must be encrypted.
- Ensure that organizations adheres to legal, regulatory and industry mandates that require specific, strong cryptographic controls for data protection.
- Prohibit the use of weak, outdated, or known-vulnerable algorithms and protocols, thereby reducing the risk of a security breach.
- System-wide cryptographic policy that provides a centralized way for administrators to configure core cryptographic like TLS, SSH, etc. so that most applications use a default, secure configuration.
- Specify when and how different types of data must be encrypted.
- Define the procedures for the entire lifecycle of cryptographic keys.
- Mandate which cryptographic algorithms are approved for use, ensuring a consistent, secure level of protection across all systems and applications.

EXAMPLE 15: ENFORCE FUTURE POLICY

```
---
- name: Enforce the 'FUTURE' system-wide cryptographic policy
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
    crypto_policies_policy: "FUTURE"
    crypto_policies_reload: true
    crypto_policies_reboot_ok: false

  tasks:
    - name: Enforce the FUTURE policy
      ansible.builtin.include_role:
        name: suse.linux_system_roles.crypto_policies
```

- `crypto_policies_policy: "FUTURE"`: is used to system-wide cryptographic settings on compatible Linux distributions. `crypto_policies_policy`is a `string`datatype and accepts values `DEFAULT`, `FUTURE`, `LEGACY` and custom policies that Administrators can define. Default is `null`.
- `crypto_policies_reload: true`: ensures that the new cryptographic policy takes effect immediately by attempting to restart or reload services that rely on the system's cryptographic libraries. `crypto_policies_reload`is a `Boolean`datatype and accepts values `true` and `false` can define. Default is `true`.
- `crypto_policies_reboot_ok: false`: dictates whether the playbook is permitted to automatically reboot the managed server after changing the cryptographic policy. `crypto_policies_reboot_ok`is a `Boolean`datatype and accepts values `true` and `false` can define. Default is `false`.

EXAMPLE 16: CONFIGURES THE DEFAULT CRYPTO POLICY LEVEL WITHOUT SHA1

```
---
- name: Manage crypto policies
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
    crypto_policies_policy: "DEFAULT:NO-SHA1"
    crypto_policies_reload: false

  tasks:
    - name: Configure default crypto policy level without SHA1
      ansible.builtin.include_role:
        name: suse.linux_system_roles.crypto_policies
```

- `crypto_policies_policy: "DEFAULT:NO-SHA1`: is used to system-wide cryptographic settings on compatible Linux distributions. `crypto_policies_policy`is a `string`datatype and accepts values `DEFAULT`, `FUTURE`, `LEGACY` and custom policies that Administrators can define. Default is `null`. In this example, you are using the `DEFAULT` policy as a starting point and then modifying it with the built-in `NO-SHA1` sub-policy.
- `crypto_policies_reload: false`: ensures that the new cryptographic policy takes effect immediately by attempting to restart or reload services that rely on the system's cryptographic libraries. `crypto_policies_reload`is a `Boolean`datatype and accepts values `true` and `false` can define. Default is `true`.

For more details about all the variables in the Crypto policies Linux system role, refer to the specific `README.md` file on the control node:

12 About the MSSQL Linux system role

The MSSQL Linux system role in Ansible automates the installation, configuration and initial deployment of Microsoft SQL Server (MSSQL) on SUSE Linux Enterprise Server 16.0 systems.

You can use this role to:

- Install the necessary SQL Server packages and tools.
- Manages the acceptance of the End-User License Agreement (EULA), which is required for installation.
- Set the required System Administrator `sa` and `mssql_password`. This is a mandatory variable for initial setup.
- Configures the TCP Port the server listens on.
- Manages TLS/SSL certificates to ensure encrypted connections to the database.
- Optimize the operating system for database performance and throughput, by applying the `mssql` Tuned profile.

EXAMPLE 17: INSTALL AND CONFIGURE BASIC MICROSOFT SQL SERVER

```
---
- name: Install and configure basic Microsoft SQL Server
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
    mssql_accept_microsoft_sql_server_standard_eula: true
    mssql_password: "{{ sa_secret_password }}"
    mssql_edition: "Developer"

  tasks:
    - name: Manage basic Microsoft SQL Server
      ansible.builtin.include_role:
```

```
        name: suse.linux_system_roles.mssql
```

- `mssql_accept_microsoft_sql_server_standard_eula`: explicitly confirms the user agrees to the Microsoft SQL Server End-User License Agreement (EULA). It is a mandatory `Boolean` datatype and accpets values `true`or `false`. It must be set to `true` in your playbook or inventory for the role to proceed with the installation of SQL Server. If set to `false`, the installation tasks will fail or be skipped. The default is empty`{}`.
- `mssql_password`: sets the password for the Administrator `sa` user. It is a mandatory `string` datatype. The password must have a minimum length of 8 characters, include uppercase and lowercase letters, base 10 digits or non-alphanumeric symbols. Default is `null`.
- `mssql_edition`: defines the specific version of Microsoft SQL Server to be installed on the host. It is a mandatory `string` datatype and accepts values `Enterprise`, `Standard`, `Web`, `Developer`, `Express`, `Evaluation` and a product key in form of `#####-#####-#####-#####-#####` where `#` is a number or letter. Default is `null`.

For more details about all the variables in the MSSQL Linux system role, refer to the specific `README.md` file on the control node:

```
/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs
```

13 About the SELinux Linux system role

The SELinux Linux system role in Ansible automates the full management and enforcement of the SELinux policy on managed nodes in a standardized and idempotent way.

You can use this role to:

- Set the global SELinux mode to `enforcing`, `permissive` or `disabled` using the `selinux_state` variable.
- Manage the state of SELinux booleans.
- Define persistent rules for file system labeling (file contexts) for custom directories or applications.
- Define security types for non-standard network ports, allowing services to listen on them without violating the policy.
- Manage the mapping between Linux user accounts and specific SELinux user identities.

EXAMPLE 18: RESET THE SELINUX CONTEXT

```
---
- name: Reset the selinux context
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
  selinux_restore_dirs:
    - /var/www/
    - /etc/

  tasks:
    - name: reset the selinux context
      ansible.builtin.include_role:
        name: suse.linux_system_roles.selinux
```

- `selinux_restore_dirs`: specifies file system paths on the managed node where the SELinux contexts should be immediately reapplied or corrected (using the equivalent of the `restorecon` utility). It is a list of strings, where each string is a file system tree where you want to run `restorecon`.

EXAMPLE 19: SET A SELINUX NETWORK PORT LABEL

```
---
- name: set a selinux port label
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
    selinux_ports:
      - ports: 8080
        proto: tcp
        setype: http_port_t
        state: present

    tasks:
      - name: set a selinux port label
          ansible.builtin.include_role:
            name: suse.linux_system_roles.selinux
```

- `selinux_ports`: manages the SELinux port labeling policy on a managed node. It is a list of dictionaries with each dictionary defining a specific port rule. `ports` are strings or a list and define port numbers to which you want to assign the SELinux label. Multiple values separated by a comma are accepted. `proto` is a `string` datatype and defines the network protocol. `setype` is a `string` datatype and defines the SELinux type (label) to assign to the port.

For more details about all the variables in the SELinux Linux system role, refer to the specific `README.md` file on the control node:

```
/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs
```

14 About the SSH Linux system role

The SSH Linux system role in Ansible automates the configuration and management of the Secure Shell (SSH) service on SUSE Linux Enterprise Server 16.0 systems.

You can use this role to:

- Enforce security best practices on the SSH daemon.
- Set default client options for all managed users.
- Deploy public keys for specific users to enable passwordless access.
- Define which users or groups are permitted to log in via SSH.

EXAMPLE 20: MANAGE SSH CLIENTS

```
---
- name: Manage ssh clients
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
      ssh_user: root
      ssh:
        Compression: true
        # wokeignore:rule=master
        ControlMaster: auto
        ControlPath: ~/.ssh/.cm%C
        Match:
```

```
          - Condition: "final all"
            GSSAPIAuthentication: true
        Host:
          - Condition: example
            Hostname: example.com
            User: user1
      ssh_ForwardX11: false

tasks:
  - name: manage SSH clients
    ansible.builtin.include_role:
      name: suse.linux_system_roles.ssh
```

- `ssh_user`: is a role specific variable used to determine the scope of the SSH client configuration being deployed. The default value is `null` or undefined, which means the role is configured to manage the global, system-wide SSH client configuration, which is located at `/etc/ssh/ssh_config` on the managed node.
- `ssh_ForwardX11`: controls whether X11 forwarding should be automatically enabled when establishing an SSH connection to a remote host. It determines if the graphical interface environment (X Window System) of the remote host should be tunneled securely back to the local client machine, allowing the user to run remote graphical applications. It is a `Boolean` datatype with values `yes` and `no`.

For more details about all the variables in the SSH Linux system role, refer to the specific `README.md` file on the control node:

```
/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs
```

15 About the systemd Linux system role

The systemd Linux system role in Ansible automates the full lifecycle management and deployment of `systemd` units and configurations SUSE Linux Enterprise Server 16.0 systems.

You can use this role to:

- Place custom or templated `systemd` unit files such as `.service`, `.timer` etc. into the correct directories `/etc/systemd/system/`.
- Control the execution status of services, allowing you to easily start, stop, restart, or reload specific units.

- Control whether units are configured to start automatically at boot or are prevented from doing so.
- Use the most aggressive method to prevent a service from ever running or being manually started by setting the unit as `masked`.

EXAMPLE 21: START AND ENABLE A SYSTEMD UNIT

```
---
- name: Deploy and start a systemd unit
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
    systemd_unit_file_templates:
      - test.service.j2
    systemd_started_units:
      - item: test.service
        user: root
      - item: test1.service
        user: user1
    systemd_enabled_units:
      - test.service
  tasks:
    - name: Manage a systemd unit
      ansible.builtin.include_role:
        name: suse.linux_system_roles.systemd
```

- `systemd_unit_file_templates`: specifies a list of Jinja2 template file names residing on the control node that should be rendered and deployed as full `systemd` unit files on the managed nodes. It is a list of strings or dictionaries. Each item specifies a Jinja2 template file located in the role's `templates/` directory. Default is an empty list `[]` or `null`.
- `systemd_started_units`: specifies which `systemd` units should be set to the started state on the managed nodes. It is a list of strings or dictionaries. Default is an empty list `[]`.
- `systemd_enabled_units`: specifies which `systemd` unit files should be enabled on the managed nodes, ensuring they start automatically at boot. It is a list of strings or dictionaries. Default is an empty list `[]`

For more details about all the variables in the systemd Linux system role, refer to the specific `README.md` file on the control node:

```
/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs
```

16 About the time synchronization Linux system role

The time synchronization Linux system role in Ansible automates the configuration and enforcement of system time synchronization on managed Linux hosts. NTP and PTP are essential standards used to synchronize computer clocks across a network. Accurate time synchronization is crucial because many critical network services rely on precise time to function correctly.

You can use this role to:

- Install and configure the preferred time synchronization service such as Chrony or NTPD.
- Set the list of reliable time servers like NTP peers and servers the system should synchronize with.
- Ensure the system uses the desired synchronization method.
- Helps ensure that systems are compliant with security requirements that mandate accurate logging and synchronized authentication, for example, Kerberos.

EXAMPLE 22: INSTALL AND CONFIGURE NTP TO SYNCHRONIZE THE SYSTEM CLOCK WITH MANAGED NODES

```
---
- name: Manage timesync
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
    timesync_ntp_servers:
      - hostname: ntp.example.com
        iburst: true
      - hostname: time.example.com
        iburst: true
      - hostname: sync.example.com
        iburst: true

  tasks:
    - name: Manage timesync
        ansible.builtin.include_role:
        name: suse.linux_system_roles.timesync
```

- `timesync_ntp_servers`: is a primary mechanism for an administrator to specify which external network time sources (NTP servers) the managed Linux host should use for clock synchronization. It is a list of dictionaries.The default is an empty list`[]`. Each dictionary

entry within the `timesync_ntp_servers`list defines a single time source and its desired configuration options. In this example, `hostname` is a required `string` datatype and `iburst` is an optional `Boolean` datatype.

EXAMPLE 23: INSTALL AND CONFIGURE LINUXPTP

```
---
- name: Manage timesync in PTP domain 0, interface eth0
  hosts: managed_nodes
  become: true
  gather_facts: true

  vars:
    timesync_ptp_domains:
      - number: 0
        interfaces: [eth0]

  tasks:
    - name: Manage linuxptp
        ansible.builtin.include_role:
        name: suse.linux_system_roles.timesync
```

- `timesync_ptp_domains`: configures Precision Time Protocol (PTP) synchronization It is a list of dictionaries. Default is an empty list `[]`. In this example, `number` is a required `string` datatype and `interfaces` are optional `list of strings` data types.

For more details about all the variables in the time synchronization Linux system role, refer to the specific `README.md` file on the control node:

```
/usr/share/ansible/collections/ansible_collections/suse/linux_system_roles/docs
```

17 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.