在虚拟机上使用原始磁盘映像部署 SLE Micro

解释

SLE Micro 提供可直接部署到虚拟机的原始映像（也称为**预构建映像**）。

原因

虚拟化部署可节省硬件资源。

工作量

读完本文大约需要 20 分钟。

目标

SLE Micro 已成功部署到虚拟机。

要求

- 已安装且正在运行 `libvirt` 和 KVM 虚拟化环境的 VM 主机服务器。
- 至少 32 GB 的磁盘空间，用于部署映像。
- （可选）一个配置媒体，例如 USB 闪存盘。

出版日期：2025 年 12 月 11 日

目录

1 关于预构建映像

预构建映像代表正在运行的操作系统，随时可供使用。它们不是通过安装程序以传统方式安装的，而是会复制到目标主机的硬盘中。本主题将介绍有关这些预构建映像的基本信息。

在首次引导时，需使用预构建映像中提供的工具对其进行配置。引导加载程序会按照第 1.2 节 “首次引导检测”中所述检测首次引导。每个映像附带默认挂载的子卷，在首次引导配置期间，这些子卷可能会发生更改。有关子卷的细节，请参见第 1.1 节 “默认分区”。

1.1 默认分区

提供的预构建映像会使用默认的分区方案，在首次引导期间，可以使用 Ignition 或 Combustion 更改该方案。

重要：根文件系统必须使用 Btrfs

如果您要对默认分区方案进行任何更改，根文件系统必须是 Btrfs。

每个映像都具有以下子卷：

```
/home
/root
/opt
/srv
/usr/local
/var
```

`/etc` 目录会挂载为 OverlayFS，其中上一级目录挂载到 `/var/lib/overlay/1/etc/`。

您可以通过 `/etc/fstab` 中的 `x-initrd.mount` 选项来识别默认挂载的子卷。其他子卷或分区必须通过 Ignition 或 Combustion 进行配置。

1.2 首次引导检测

部署配置只在首次引导时运行。为了区分首次引导和后续引导，完成首次引导后，系统会创建标志文件 `/boot/writable/firstboot_happened`。如果文件系统中没有该文件，则会将 `ignition.firstboot` 属性传递给内核命令行，因而触发 Ignition 和 Combustion 的运行（在 initrd 中）。完成首次引导后，系统便会创建 `/boot/writable/firstboot_happened` 标志文件。

注意：系统始终会创建该标志文件

即使由于配置文件错误或缺失导致配置失败，系统也会创建 `/boot/writable/firstboot_happened` 标志文件。

1.2.1 在后续引导时强制重配置系统

如果您需要在发生首次引导后重配置系统，可以在后续引导时强制重配置。此处可以采取两种做法。

- 可以将 `ignition.firstboot=1` 属性传递给内核命令行。
- 可以删除标志文件 `/boot/writable/firstboot_happened`。

2 准备配置设备

重要：SSH 登录

默认情况下，SLE Micro 中仅允许使用 SSH 密钥进行 `root` SSH 登录。我们建议在部署过程中创建一个非特权用户用于访问已安装的系统。可以在首次引导时使用 Combustion 或 Ignition 工具创建非特权用户帐户。在系统部署期间创建的非特权用户还可用于访问 Cockpit Web 界面。

要准备配置设备，请执行以下步骤：

过程 1： **准备配置设备**

1. 将磁盘格式化为 SLE Micro 支持的任何文件系统：Ext3、Ext4 等：

```
> sudo mkfs.ext4 /dev/sdY
```

2. 将设备标签设置为 `ignition`（使用 Ignition 或 时）或 `combustion`combustion（仅使用 Combustion 时）。如果需要（例如在 Windows 主机上），请为标签使用大写字母。要为设备设置标签，请运行：

```
> sudo e2label /dev/sdY ignition
```

可以使用您的虚拟化系统或硬件支持的任何类型的配置存储媒体：ISO 映像、USB 闪存盘等

3. 挂载设备：

```
> sudo mount /dev/sdY /mnt
```

4. 创建第 2.1.1.1 节 “`config.ign`”或第 2.2 节 “使用 Combustion 配置 SLE Micro 部署”中所述的目录结构，具体取决于使用的配置工具：

```
> sudo mkdir  /mnt/ignition/
```

或：

```
> sudo mkdir -p /mnt/combustion/
```

5. 准备 Ignition 或 Combustion 所用配置的所有元素

2.1 使用 Ignition 配置 SLE Micro 部署

Ignition (https://coreos.github.io/ignition/) 是一种预配工具，可让您在首次引导时根据您的具体要求配置系统。

2.1.1 Ignition 的工作原理

首次引导系统时，Ignition 将作为 `initramfs` 的一部分加载，并在特定的目录中（在 USB 闪存盘上，或者您可以提供 URL）搜索配置文件。所有更改都是在内核从临时文件系统切换到实际根文件系统前（即在 `switch_root` 命令发出前）进行的。

Ignition 使用名为 `config.ign` 的 JSON 格式配置文件。您可以手动编写配置，也可以使用 https://ignite.opensuse.org 上的 Fuel Ignition Web 应用程序生成配置。

重要

Fuel Ignition 尚未涵盖完整的 Ignition 词汇，生成的 JSON 文件可能需要进行额外的手动调整。

2.1.1.1 `config.ign`

配置文件 `config.ign` 必须位于配置媒体（例如，标签为 `ignition` 的 USB 记忆棒）的 `ignition` 子目录中。目录结构必须如下所示：

```
<root directory>
└── ignition
    └── config.ign
```

提示

要通过 Ignition 配置创建磁盘映像，可以使用 https://ignite.opensuse.org 上的 Fuel Ignition Web 应用程序。

`config.ign` 包含多种数据类型：对象、字符串、整数、布尔值和对象列表。如需完整规范，请参见 Ignition specification v3.3.0 (https://coreos.github.io/ignition/configuration-v3_3/)。

`version` 属性是必需的，在 SLE Micro 中，其值必须设置为 `3.3.0` 或任何更低版本。否则，Ignition 将会失败。

要以 root 身份登录到系统，必须至少包含 root 的口令。但建议通过 SSH 密钥建立访问权限。要配置口令，请务必使用安全口令。如果您使用随机生成的口令，请至少包含 10 个字符。如果您要手动创建口令，请包含 10 个以上的字符，并结合使用大写与小写字母和数字。

2.1.2 Ignition 配置示例

2.1.2.1 配置示例

本部分提供内置 JSON 格式的 Ignition 配置的几个示例。

重要

第 1.1 节 “默认分区”列出了在运行预构建的映像时默认挂载的子卷。如果您要在默认未挂载的子卷上添加新用户或修改任何文件，则需要先声明此类子卷，以便将其挂载。有关挂载文件系统的详细信息，请参见第 2.1.2.1.1.3 节 “filesystems 属性”。

注意：version 属性为必要属性

每个 config.fcc 都必须包含版本 1.4.0 或更低版本，该版本随后会转换为相应的 Ignition 规范。

2.1.2.1.1 储存配置

storage 属性用于配置分区、RAID，定义文件系统，创建文件等。要定义分区，请使用 disks 属性。filesystems 属性用于格式化分区和定义特定分区的挂载点。files 属性可用于在文件系统中创建文件。后续章节中将介绍上述每个属性。

2.1.2.1.1.1 disks 属性

disks 属性是设备列表，可用于定义这些设备上的分区。disks 属性必须至少包含一个 device，其他属性为可选属性。以下示例使用单个虚拟设备，并将磁盘划分为四个分区：

```
{
```

```
  "ignition": {
    "version": "3.0.0"
  },
  "storage": {
    "disks": [
      {
        "device": "/dev/vda",
        "partitions": [
          {
            "label": "root",
            "number": 1,
            "typeGuid": "4F68BCE3-E8CD-4DB1-96E7-FBCAF984B709"
          },
          {
            "label": "boot",
            "number": 2,
            "typeGuid": "BC13C2FF-59E6-4262-A352-B275FD6F7172"
          },
          {
            "label": "swap",
            "number": 3,
            "typeGuid": "0657FD6D-A4AB-43C4-84E5-0933C84B4F4F"
          },
          {
            "label": "home",
            "number": 4,
            "typeGuid": "933AC7E1-2EB4-4F13-B844-0E14E2AEF915"
          }
        ],
        "wipeTable": true
      }
    ]
  }
}
```

2.1.2.1.1.2 `raid` 属性

`raid` 是 RAID 阵列列表。`raid` 的下列属性为必要属性：

level

特定 RAID 阵列的级别（线性、raid0、raid1、raid2、raid3、raid4、raid5、raid6）

devices

阵列中设备的列表，通过绝对路径引用这些设备

name

将用于 md 设备的名称

例如：

```
{
  "ignition": {
    "version": "3.0.0"
  },
  "storage": {
    "raid": [
      {
        "devices": [
          "/dev/sda",
          "/dev/sdb"
        ],
        "level": "raid1",
        "name": "system"
      }
    ]
  }
}
```

2.1.2.1.1.3 `filesystems` 属性

`filesystems` 必须包含以下属性：

device

设备的绝对路径，如果是物理磁盘，通常为 `/dev/sda`

format

文件系统格式（Btrfs、Ext4、xfs、vfat 或 swap）

注意

对于 SLE Micro，`root` 文件系统必须为 Btrfs 格式。

下面的示例演示如何使用 `filesystems` 属性。`/opt` 目录将挂载到 `/dev/sda1` 分区，该分区为 Btrfs 格式。系统将不会擦除该设备。

例如：

```
{
  "ignition": {
    "version": "3.0.0"
  },
  "storage": {
    "filesystems": [
      {
        "device": "/dev/sda1",
        "format": "btrfs",
        "path": "/opt",
        "wipeFilesystem": false
      }
    ]
  }
}
```

普通用户的主目录通常位于 `/home/USER_NAME` 目录中。由于 `/home` 默认不会挂载到 initrd 中，因此必须明确定义挂载，才能成功创建用户：

```
{
  "ignition": {
    "version": "3.1.0"
  },
  "passwd": {
    "users": [
      {
        "name": "root",
        "passwordHash": "PASSWORD_HASH",
        "sshAuthorizedKeys": [
          "ssh-rsa SSH_KEY_HASH"
```

```
        ]
      }
    ]
  },
  "storage": {
    "filesystems": [
      {
        "device": "/dev/sda3",
        "format": "btrfs",
        "mountOptions": [
          "subvol=/@/home"
        ],
        "path": "/home",
        "wipeFilesystem": false
      }
    ]
  }
}
```

2.1.2.1.1.4 `files` 属性

您可以使用 `files` 属性在计算机上创建任何文件。请注意，要在默认的分区方案之外创建文件，需使用 `filesystems` 属性定义目录。

在下面的示例中，将使用 `files` 属性创建一个主机名。将创建文件 `/etc/hostname`，其中包含 **sl-micro1** 主机名：

重要

请记住，JSON 接受十进制数格式的文件模式，例如 `420`。

JSON：

```
{
  "ignition": {
    "version": "3.0.0"
  },
```

```
  "storage": {
    "files": [
      {
        "overwrite": true,
        "path": "/etc/hostname",
        "contents": {
          "source": "data:,sl-micro1"
        },
        "mode": 420
      }
    ]
  }
}
```

2.1.2.1.1.5 directories 属性

directories 属性是将在文件系统中创建的目录列表。directories 属性必须至少包含一个 path 属性。

例如：

```
{
  "ignition": {
    "version": "3.0.0"
  },
  "storage": {
    "directories": [
      {
        "path": "/home/tux",
        "user": {
          "name": "tux"
        }
      }
    ]
  }
}
```

2.1.2.1.2 用户管理

passwd 属性用于添加用户。由于某些服务（例如 Cockpit）要求使用非 root 用户身份登录，因此请在此处至少定义一个非特权用户。或者，您可以按第 5.3 节 “添加用户” 中所述在正在运行的系统中创建此类用户。

要登录系统，请创建 root 和一个普通用户，并设置他们的口令。您需要对口令进行哈希处理，例如使用 **openssl** 命令来处理：

```
openssl passwd -6
```

该命令会为您选择的口令创建哈希。使用此哈希作为 password_hash 属性的值。

例如：

```
{
  "ignition": {
    "version": "3.0.0"
  },
  "passwd": {
    "users": [
      {
        "name": "root",
        "passwordHash": "PASSWORD_HASH",
        "sshAuthorizedKeys": [
          "ssh-rsa SSH_KEY_HASH USER@HOST"
        ]
      }
    ]
  }
}
```

users 属性必须至少包含一个 name 属性。ssh_authorized_keys 是用户的 SSH 密钥列表。

2.1.2.1.3 启用 systemd 服务

您可以通过在 systemd 属性中指定 systemd 服务来启用相应服务。

例如：

```
{
  "ignition": {
    "version": "3.0.0"
  },
  "systemd": {
    "units": [
      {
        "enabled": true,
        "name": "sshd.service"
      }
    ]
  }
}
```

2.2 使用 Combustion 配置 SLE Micro 部署

Combustion 是一种 dracut 模块，可用于在首次引导时配置系统。您可以使用 Combustion 来更改默认分区、设置用户口令、创建文件、安装软件包等。

2.2.1 Combustion 的工作原理

系统将在 `ignition.firstboot` 参数传递给内核命令行后调用 Combustion。Combustion 会读取提供的文件（名为 `script`）并执行其中的命令，以对文件系统进行更改。如果 `script` 中包含网络标志，Combustion 会尝试配置网络。挂载 `/sysroot` 后，Combustion 会尝试激活 `/etc/fstab` 中的所有挂载点，然后调用 **`transactional-update`** 来应用其他更改，例如设置 `root` 口令或安装软件包。

配置文件 `script` 必须位于标记为 `combustion` 的配置媒体的 `combustion` 子目录中。目录结构必须如下所示：

```
<root directory>
└── combustion
    └── script
```

```
    └── other files
```

提示：搭配使用 Combustion 与 Ignition

Combustion 可与 Ignition 搭配使用。如果您要将它们搭配使用，请将配置媒体标记为 `ignition`，并在目录结构中添加包含 `config.ign` 的 `ignition` 目录，如下所示：

```
<root directory>
└── combustion
    └── script
    └── other files
└── ignition
    └── config.ign
```

在此情况下，Ignition 会先于 Combustion 运行。

2.2.2 Combustion 配置示例

2.2.2.1 `script` 配置文件

`script` 配置文件是一组由 Combustion 在 **`transactional-update`** 外壳中分析并执行的命令。本文提供了 Combustion 执行的配置任务的一些示例。

重要：包含解释器声明

`script` 文件由外壳解释，所以请务必在文件的第一行以解释器声明开头。例如，对于 Bash：

```
#!/bin/bash
```

为了登录系统，请至少包含 `root` 口令，但建议使用 SSH 密钥建立身份验证。如果您需要使用 `root` 口令，请务必配置安全口令。对于随机生成的口令，请至少包含 10 个字符。如果您要手动创建口令，请包含 10 个以上的字符，并结合使用大写与小写字母和数字。

2.2.2.1.1 网络配置

要在首次引导期间配置并使用网络连接，请在 `script` 中添加以下语句：

```
# combustion: network
```

使用此语句会将 `rd.neednet=1` 参数传递给 dracut。网络配置默认设为使用 DHCP。如果需要使用不同的网络配置，请按照第 2.2.2.1.2 节 “在 initramfs 中进行修改”中所述操作。

如果不使用该语句，将不会为系统配置任何网络连接。

2.2.2.1.2 在 initramfs 中进行修改

您可能需要对 initramfs 环境进行更改，例如，将 NetworkManager 的自定义网络配置写入 `/etc/NetworkManager/system-connections/` 中。要执行此操作，请使用 `prepare` 语句。

例如，要使用静态 IP 地址创建连接并配置 DNS，请执行以下操作：

```
#!/bin/bash
# combustion: network prepare
set -euxo pipefail

nm_config() {
  umask 077 # Required for NM config
  mkdir -p /etc/NetworkManager/system-connections/
  cat >/etc/NetworkManager/system-connections/static.nmconnection <<-EOF
  [connection]
  id=static
  type=ethernet
  autoconnect=true

  [ipv4]
  method=manual
  dns=192.168.100.1
  address1=192.168.100.42/24,192.168.100.1
EOF
}
```

```
if [ "${1-}" = "--prepare" ]; then
  nm_config # Configure NM in the initrd
  exit 0
fi

# Redirect output to the console
exec > >(exec tee -a /dev/tty0) 2>&1

  nm_config # Configure NM in the system
  curl example.com
# Leave a marker
echo "Configured with combustion" > /etc/issue.d/combustion
```

2.2.2.1.3 分区

提供的 SLE Micro 原始映像会使用第 1.1 节 “默认分区”中所述的默认分区方案。您可能希望使用不同的分区方式。下面一组示例代码段会将 `/home` 移至另一个分区。

注意：在快照中包含的目录外部进行更改

以下脚本会执行快照中未包含的更改。如果脚本失败且快照被丢弃，那么某些更改将仍然可见且无法还原，例如对 `/dev/vdb` 设备的更改。

以下代码段会在 `/dev/vdb` 设备上创建仅含一个分区的 GPT 分区方案：

```
sfdisk /dev/vdb <<EOF
label: gpt
type=linux
EOF

partition=/dev/vdb1
```

该分区为 Btrfs 格式：

```
wipefs --all ${partition}
mkfs.btrfs ${partition}
```

以下代码段会将 `/home` 中可能包含的内容移到新的 `/home` 文件夹位置：

```
mount /home
mount ${partition} /mnt
rsync -aAXP /home/ /mnt/
umount /home /mnt
```

下面的代码段会去除 `/etc/fstab` 中的旧项并创建新项：

```
awk -i inplace '$2 != "/home"' /etc/fstab
echo "$(blkid -o export ${partition} | grep ^UUID=) /home btrfs defaults 0 0"
 >>/etc/fstab
```

2.2.2.1.4 创建新用户

由于某些服务（例如 Cockpit）要求使用非 root 用户身份登录，因此请在此处至少定义一个非特权用户。或者，您可以按第 5.3 节 “添加用户”中所述在正在运行的系统中创建此类用户。

要添加新用户帐户，请先创建一个代表用户口令的哈希字符串。使用 **`openssl passwd -6`** 命令。

获取口令哈希后，在 `script` 中添加以下几行内容：

```
mount /home
useradd -m EXAMPLE_USER
echo 'EXAMPLE_USER:PASSWORD_HASH' | chpasswd -e
```

2.2.2.1.5 设置 root 的口令

在设置 `root` 口令前，请先生成口令的哈希，例如，使用 **`openssl passwd -6`** 来生成。要设置口令，请在 `script` 中添加以下几行内容：

```
echo 'root:PASSWORD_HASH' | chpasswd -e
```

2.2.2.1.6 添加 SSH 密钥

以下代码段会创建用于存储 `root` 的 SSH 密钥的目录，然后将位于配置设备上的公共 SSH 密钥复制到 `authorized_keys` 文件中。

```
mkdir -pm700 /root/.ssh/
cat id_rsa_new.pub >> /root/.ssh/authorized_keys
```

注意

如果您需要通过 SSH 进行远程登录，则必须启用 SSH 服务。有关细节，请参见第 2.2.2.1.7 节 “启用服务”。

2.2.2.1.7 启用服务

要启用系统服务（例如 SSH 服务），请将下面一行添加到 `script` 中：

```
systemctl enable sshd.service
```

2.2.2.1.8 安装软件包

重要：可能需要注册您的系统并连接网络

由于您可能需要额外订阅特定的软件包，因此可能需事先注册系统。此外，安装额外的软件包可能还需要有网络连接。

在首次引导配置期间，可以在系统上安装额外的软件包。例如，可以通过添加以下命令安装 `vim` 编辑器：

```
zypper --non-interactive install vim-small
```

注意

请注意，在配置完毕并引导到配置的系统前，您将无法使用 **`zypper`**。如果想要在稍后进行更改，则必须使用 **`transactional-update`** 命令创建已更改快照。

3 准备虚拟机

本节介绍如何准备新虚拟机，以及在该虚拟机上部署 SLE Micro 时需执行哪些步骤。

1. 在要用来运行虚拟化 SLE Micro 的 VM 主机服务器上下载 SLE Micro 磁盘映像。
2. 启动虚拟机管理器，然后选择文件 › 新建虚拟机。
3. 选择导入现有磁盘映像。单击前进确认。
4. 指定之前下载的 SLE Micro 磁盘映像的路径以及要部署的 Linux 操作系统类型（例如 `Generic Linux 2020`）。单击前进确认。
5. 指定要分配给 SLE Micro 虚拟机的内存容量和处理器数量，然后单击前进确认。
6. 指定虚拟机名称和要使用的网络。
7. 如果要部署加密 SLE Micro 映像，另外还需执行以下步骤：
    a. 启用在安装前自定义配置，然后单击完成确定。
    b. 单击左侧菜单中的概览，将引导方法从 BIOS 更改为 UEFI，以实现安全引导。单击应用进行确认。

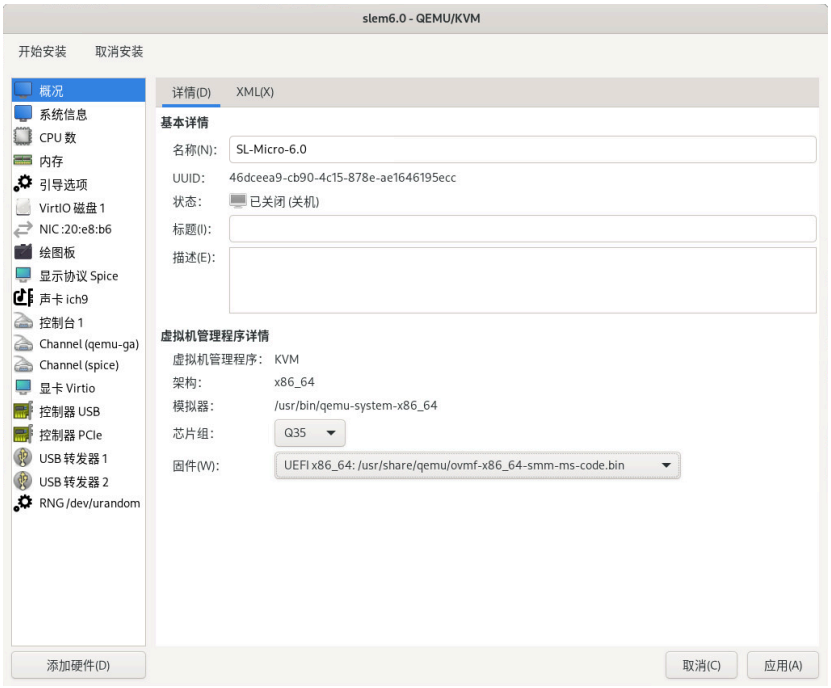

图 1： 为加密的 SLE MICRO 映像设置 UEFI 固件

c. 添加可信平台模块 (TPM) 设备。单击添加硬件，从左侧菜单中选择 TPM，然后选择模拟类型。

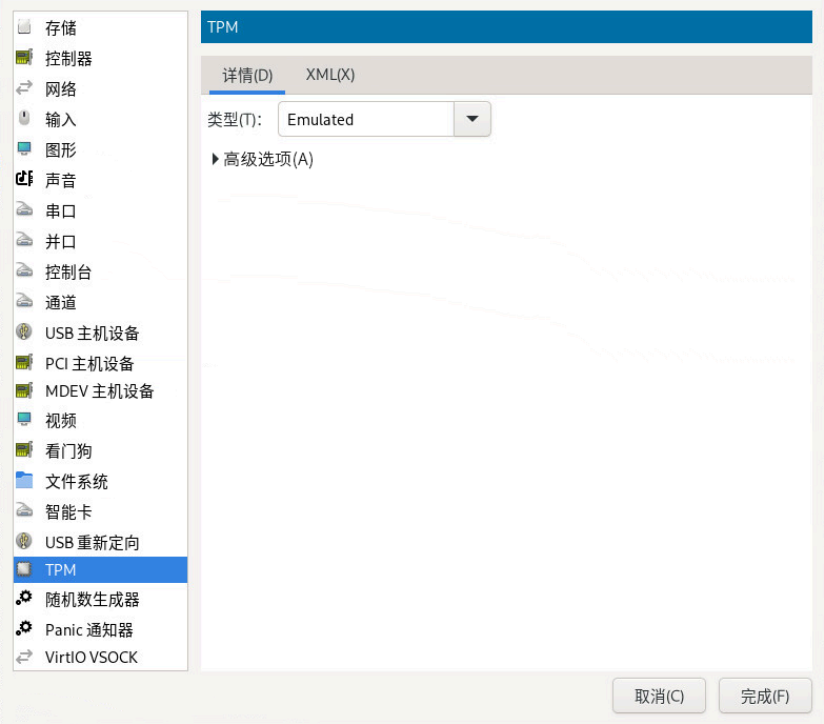

图 2： 添加模拟的 TPM 设备

单击完成确认，然后在顶部菜单中单击开始安装，启动 SLE Micro 部署过程。

4 使用 JeOS Firstboot 进行配置

首次引导 SLE Micro 时，如果未提供任何配置设备，**JeOS Firstboot** 可用于对系统执行最低限度的配置。如果需要更好地控制部署过程，请使用具有 Ignition 或 Combustion 配置的配置设备。有关详细信息，请参见第 2.1 节 “使用 Ignition 配置 SLE Micro 部署”和第 2.2 节 “使用 Combustion 配置 SLE Micro 部署”。

要使用 **JeOS Firstboot** 配置系统，请按以下步骤操作：

1. JeOS Firstboot 会显示欢迎屏幕。单击 `Enter` 确认。
2. 在接下来的屏幕中，选择键盘，确认许可协议，并选择时区。
3. 在输入 root 密码对话框窗口中，输入 `root` 的密码并确认。

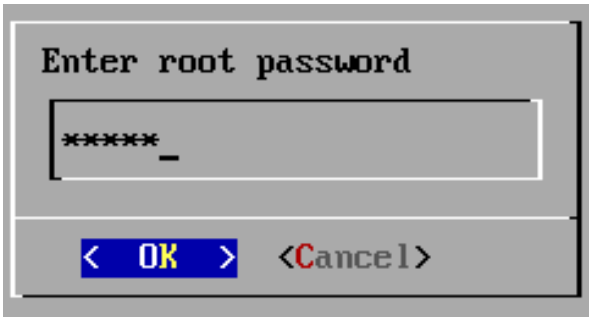

图 3： 输入 ROOT 密码

4. 对于加密部署，JeOS Firstboot 会执行以下操作：

   - 提示输入新的通行口令，用来取代默认的通行口令。
   - 生成新的 LUKS 密钥，并重新加密分区。
   - 为 LUKS 报头添加次要密钥槽，并将其封装到 TPM 设备中。

   如果您要部署的是加密映像，请执行以下步骤：

   a. 选择所需的保护方法，并单击确定确认。

   b. 输入 LUKS 加密的恢复口令，然后再输入一遍。根文件系统开始重新加密。

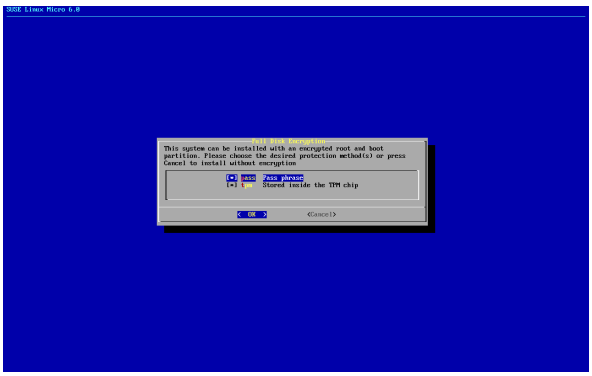

图 4： 选择加密方法

5. 成功部署后，按第 5.4 节 “从 CLI 注册 SLE Micro”中所述注册系统并创建非特权用户。

5 部署后步骤

5.1 扩展加密的磁盘映像

SLE Micro 的加密原始磁盘映像不会自动扩展到磁盘的全部容量。此过程概述了将它们扩展到所需大小的步骤。

过程 2： 扩展加密的磁盘映像

1. 使用 **qemu-img** 命令将磁盘映像增至所需大小。
2. 使用 **parted** 命令将 LUKS 设备所在的分区（例如分区 3）调整为所需大小。
3. 运行 **cryptsetup resize luks** 命令：当系统要求输入通行口令时，请输入相应口令以调整加密设备的大小。
4. 运行 **transactional-update shell** 命令在当前磁盘快照中打开读写外壳，然后将 Btrfs 文件系统调整为所需大小。例如：

   ```
   # btrfs fi resize max /
   ```

5. 运行 **exit** 退出外壳，然后运行 **reboot** 重引导系统。

5.2 重新加密已加密系统

警告：系统未受保护

系统目前未受保护，因此，在完成磁盘重新加密之前，不要在其中存储任何敏感数据。

注意：如果您是使用 JeOS Firstboot 部署系统的，则不需要执行此步骤

JeOS Firstboot 会在部署阶段提示输入新通行口令。您输入通行口令后，系统会自动重新加密，因此您无需执行其他操作。

SLE Micro 加密映像附带默认的 LUKS 通行口令。要保护系统的安全，请务必在部署系统后更改该通行口令。为此，请执行以下操作。请在同一外壳会话中执行这些步骤。

1. 将所需的函数导入外壳中：

```
# source /usr/share/fde/luks
```

2. 识别底层 LUKS 设备并定义其他所使用的变量：

```
# luks_name=$(expr "`df --output=source / | grep /dev/`" :
      ".*/\(.*\)")
```

和

```
# luks_dev=$(luks_get_underlying_device "$luks_name")
```

3. 创建一个用来存储默认通行口令 **1234** 的密钥文件，以及一个包含新通行口令的密钥文件。

4. 更改恢复口令：

```
# cryptsetup luksChangeKey --key-filePATH_TO_DEFAULT --pbkdf pbkdf2
 "${luks_dev}" PATH_TO_NEW
```

`PATH_TO_DEFAULT` 是包含默认通行口令的密钥文件的路径。`PATH_TO_NEW` 是包含新通行口令的密钥文件的路径。

5. 重新加密 LUKS 设备：

```
# cryptsetup reencrypt --key-filePATH_TO_NEW ${luks_dev}
```

6. 创建一个新的随机密钥并将其封装到 TPM 中：

```
> sudo fdectl regenerate-key --passfile PATH_TO_NEW
```

7. 去除您在步骤 3 中创建的两个密钥文件。

8. 运行以下命令更新 `grub.cfg` 文件：

```
> sudo transactional-update grub.cfg
```

9. 重新启动系统。

5.3 添加用户

由于 SLE Micro 要求使用非特权用户身份通过 SSH 登录系统或访问 Cockpit，因此您需要创建这样的帐户。

如果已在 Ignition 或 Combustion 中定义了非特权用户，则可以不执行此步骤。如果您是使用 JeOS Firstboot 部署系统的，那么只需设置 `root` 口令，并且需要手动创建非特权帐户，如下所述：

1. 如下所示运行 **useradd** 命令：

```
# useradd -m USER_NAME
```

2. 为该帐户设置口令：

```
# passwdUSER_NAME
```

3. 根据需要将用户添加到 `wheel` 组：

```
# usermod -aG wheelUSER_NAME
```

5.4 从 CLI 注册 SLE Micro

部署成功后，需要注册系统以获得技术支持并接收更新。可以使用 **transactional-update register** 命令从命令行注册系统。

要在 SUSE Customer Center 中注册 SLE Micro，请执行以下操作：

1. 按如下方式运行 **transactional-update register**：

```
# transactional-update register -rREGISTRATION_CODE -e EMAIL_ADDRESS
```

要在本地注册服务器中注册，另请提供该服务器的 URL：

```
# transactional-update register -rREGISTRATION_CODE -e EMAIL_ADDRESS \
 --url "https://suse_register.example.com/"
```

将 REGISTRATION_CODE 替换为随您的 SLE Micro 一起收到的注册码。将 EMAIL_ADDRESS 替换为与您或贵组织管理订阅时所用 SUSE 帐户关联的电子邮件地址。

2. 重引导您的系统以切换到最新的快照。
3. SLE Micro 现已注册完毕。

注意：其他注册选项

如果所需的信息超出了本节的范畴，请使用 **SUSEConnect --help** 查看内嵌文档。

6 法律声明

A GNU 自由文档许可证

0. 导言

此许可证的目的是赋予手册、教科书或其他功能性的和有用的文档以“自由”：即保证每个人都有复制和再分发此类文档的有效自由，无论是否进行修改，也无论将其用于商业或非商业用途。其次，此许可证为作者和出版者保留了因工作获得声誉但不视为对他人所做修改负责的方式。

本许可证是一种“非盈利版权”，这意味着从该文档衍生的作品也必须是以同一方式自由的。它补充了 GNU 通用公共许可证（为自由软件设计的非盈利版权许可证）。

我们设计此许可证旨在将其用于免费软件的手册，因为免费软件需要自由文档：免费程序所附手册应具有与软件本身同样的自由。但是此许可证不限于软件手册；它可以用于任何文本作品，无论主题如何或它是否作为印刷书籍出版。建议本许可证主要用于目的是指导或参考的作品。

1. 适用性和定义

此许可证适用的对象：由版权所有者在其中明确声明可按照此许可证条款以任何媒体分发的任何手册和其他作品。此类声明授予在此处所述的条款和条件下使用该作品的全球无限期无版权许可证。下述“文档”指任何此类手册或作品。任何公众成员都是一个被许可人，以下称为“您”。如果您以需要版权法许可的任何方式复制、修改或分发该作品，则表示您接受该许可证。

该文档的“修改版本”表示包含该文档或其一部分（或者逐字复制或者有修改和/或翻译为另一语言）的任何作品。

“次要章节”是该文档的命名附录或扉页章节，专门讲述该文档的出版者或作者与该文档整个主题（或相关问题）的关系，不包含与整个主题相关的内容。（因此，如果该文档是数学课本的一部分，则辅助部分可能不说明任何数学问题。）这种关系可以是与主题或相关问题的历史联系，或与它们相关的法律、商业、哲学、伦理或政治地位。

在该文档基于此许可证项发布的声明中，“固定章节”是将其标题指定为固定章节标题的一些辅助章节。如果一个章节不适用上述辅助章节的定义，则不允许将其指定为固定章节。该文档可能不包含固定章节。如果该文档不标识任何固定章节，则表示没有固定章节。

在该文档基于此许可证项发布的声明中，“封页文本”是作为封面文本或封底文本列出的简短文本段落。封面文本最多 5 个单词，封底文本最多 25 个单词。

文档的“透明”副本是一个机器可读的副本，使用公众可以得到其规范的格式表达，这样的副本适合于使用通用文本编辑器、（对于像素构成的图像）通用绘图程序、（对于绘制的图形）广泛使用的绘画编辑器直接修改文档，也适用于输入到文本格式处理程序或自动翻译成各种适于适用于输入到文本格式处理程序的格式。一个用其他透明文件格式表示的副本，如果该格式的标记（或缺少标记）已经构成了对读者的后续修改的障碍，那么就是不透明的。表示实质性数量的文本的图像格式都是不透明的。不“透明”的副本称为“不透明”。

适于作为透明副本的格式的示例有：没有标记的纯 ASCII 文本、Texinfo 输入格式、LaTeX 输入格式、使用公共可用 DTD 的 SGML 或 XML，符合标准的简单 HTML、可以人为修改的 PostScript 或 PDF。透明图像格式的示例有 PNG、XCF 和 JPG。不透明的格式包括：仅可以被私有版权的字处理软件使用的私有版权格式、所用的 DTD 和/或处理工具不是广泛可用的的 SGML 或 XML、机器生成的 HTML、一些字处理器生成的只用于输出目的的 PostScript 或 PDF。

对于印刷书籍，“扉页”就是扉页本身以及随后的一些用于补充的页，显然本许可资料需要出现在扉页上。对于那些没有扉页的作品形式，“扉页”代表接近作品最突出标题的、在文本正文之前的文本。

“命名为 XYZ”的章节表示文档的一个特定的子单元，其标题就是 XYZ 或在括号中包含 XYZ 且后跟 XYZ 的其他语言翻译文本。（这里 XYZ 代表下面提及的特定章节名称，比如“致谢”、“题献”、“签名”或“历史”。）要在修改文档时对这类章节“保留标题”就是依据此定义保持这样一个“命名为 XYZ”的章节。

文档可能在文档遵照此许可证的声明后面包含免责声明。这些免责声明应作为参考信息包含在此许可证中，但是只能将其视作免责声明：这些免责声明暗指的任何其他含义均无效，且对此许可证的含义不产生任何影响。

2. 逐字复制

您可以用任何媒体复制并分发文档，无论是出于商业还是非商业目的，只要保证此许可证、版权声明和声称此许可证应用于文档的声明都完整地、无任何附加条件地存在于所有副本中。不能使用任何技术手段阻碍或控制您制作或发布的副本的阅读或再次复制。不过您可以在副本交易中得到报酬。如果发布足够多的副本，则您必须遵循下面第三节中的条件。

您也可以在如上的条件下出租副本和向公众放映副本。

3. 大量复制

如果您出版的文档印刷版副本（或是有印制封页的其他媒体副本）多于 100 份，而文档的许可证声明中要求有封页文本，则您必须将它清晰地置于封页之上，封面文本在封面上，封底文本在封底上。封面和封底上还必须标明您是这些副本的出版者。封面必须同等显著地完整展现标题的所有文字。您可以在封页上加入其他资料。改动仅限于封页的复制，只要保持文档的标题不变并满足这些条件，可以在其他方面被视为逐字复制。

如果需要加上的文本对于封面或封底过多，无法明显地表示，您应该在封页上列出前面的（在合理的前提下尽量多），把其他的放在邻近的页面上。

如果您出版或分发了超过 100 份文档的不透明副本，则必须在每个不透明副本中包含一份计算机可读的透明副本，或是在每个不透明副本中给出一个计算机网络地址，通过这个地址，网络公共用户可以使用标准网络协议下载文档的无任何附加资料的完整透明副本。如果您选择后者，则必须在开始大量分发非透明副本的时候采用相当谨慎的步骤，保证透明副本在其所给出的位置在（直接或通过代理和零售商）分发最后一次该版本的非透明副本的时间之后一年之内始终是有效的。

在重新大量发布副本之前，请您（但不是必须）与文档的作者联系，以便他们可以有机会向您提供文档的更新版本。

4. 修改

在上述第 2、3 节的条件下，您可以复制和分发文档的修改版本，前提是严格按照此许可证发布修改后的文档，将修改版本用作文档，从而允许任何拥有此修改版副本的人执行分发或修改。另外，在修改版中，您需要做到如下几点：

A. 用于与文档以及以前各个版本（如果有，应该列在文档的“历史”章节中）显著不同的扉页（和封页，如果有）。如果那个版本的原始发行者允许的话，您可以使用和以前版本相同的标题。

B. 与作者一样，在扉页上列出承担修改版本中的修改的作者责任的一个或多个人或实体和至少五个文档的原作者（如果原作者不足五个就全部列出），除非他们免除了您的这个责任。

C. 与原来的发行者一样，在扉页上列出修改版的发行者的姓名。

D. 保持该文档的全部版权声明不变。

E. 在与其他版权声明邻近的位置加入恰当的针对您的修改的版权声明。

F. 在紧接着版权声明的位置加入许可声明，按照下面附录中给出的形式，以本许可证给公众授于是用修订版本的权利。

G. 保持原文档的许可声明中的全部不可变章节、封面文字和封底文字的声明不变。

H. 包含一份未作任何修改的本协议的副本。

I. 保持命名为“历史”的章节不变，保持它的标题不变，并在其中加入一项，至少声明扉页上的已修改版本的标题、年份、新作者和出版者。如果文档中没有命名为“历史”章节，则请新建它，并加入一项以声明原文档扉页上所列的标题、年份、作者与出版者，再在其后加入如上所说的描述修改版本的项。

J. 如果文档中有用于公共用户访问的文档透明副本的网址，则保持网址不变，并同样提供它所基于的以前文档版本的网址。这些网址可以放在“历史”章节。您可以不给出那些在原文档发行之前已经发行至少四年的版本给出的网址，或者该版本的发行者授权不列出网址。

K. 对于任何命名为“致谢”或“题献”的章节，保持其标题不变，并保持其全部内容以及对每位贡献者致谢和/或题献的语气不变。

L. 保持文档的所有固定章节不变，不改变它们的标题和内容。章节的编号或相当的内容不被认为是章节标题的一部分。

M. 删除命名为“签名”的章节。这样的章节不可以被包含在修改后的版本中。

N. 不要把任何现有章节重命名为“签名”或与任何不可变章节相冲突的标题。

O. 保持任何免责声明不变。

如果修改版本加入了新的符合次要章节定义的引言或附录章节，并且不含有从原文档中复制的内容，您可以选择将其标记为固定。如果需要这样做，则将它们的标题加入修改版本许可声明的不可变章节列表之中。这些标题必须和其他章节的标题相区分。

您可以加入一个命名为“签名”的章节，只要它只包含对您的修改版本由不同的各方给出的签名，例如书评或是声明文本已经被一个组织认定为一个标准的权威定义。

您可以加入一个最多 5 个字的段落作为封面文本和一个最多 25 个字的段落作为封底文本，将它们加入修改版本的封页文本列表末端。一个实体只可以添加（或编排）一段封面和一段封底文本。如果原文档已经为该封页（封面或封底）包含了封页文本，由您或您所代表的实体先前加入或排列的文本，不能再新加入一个，但您可以在原来的发行者的明确许可下替换掉原来的那个。

作者和发行者不能通过本许可证授权公众使用他们的名字推荐或暗示认可任何一个修改版本。

5. 组合文档

遵照第 4 节所说的修改版本的规定，您以将文档和其他文档合并并以本许可证发布，只要您在合并结果中包含原文档的所有不可变章节，对它们不加以任何改动，并在合并结果的许可声明中将它们全部列为不可变章节，而且维持原作者的免责声明不变。

合并作品仅需要包含一份此许可证，多个相同的固定章节可以由一个副本取代。如果有多个名称相同但内容不同的固定章节，通过在章节名称后面的括号中加上原作者或出版者的姓名（如果已知）来加以区别，或者使用唯一编号加以区别。并对合并作品许可声明中的固定章节列表中的章节标题做相同的调整。

在合并过程中，必须合并不同原始文档中任何命名为“历史”的章节，从而形成新的命名为“历史”的章节；类似地，还要合并命名为“致谢”和“题献”的章节。必须删除所有命名为“签名”的章节。

6. 文档的合集

您可以制作一个文档和其他文档的合集，在本许可证下发布，并在合集中将不同文档中的多个本许可证的副本以一个单独的副本来代替，只要您在文档的其他方面遵循本许可证的逐字复制的条款即可。

您可以从一个这样的合集中提取一个单独的文档，并将它在本许可证下单独发布，只要您想这个提取出的文档中加入一份本许可证的副本，并在文档的其他方面遵循本许可证的逐字复制的原则。

7. 独立作品的聚合体

将文档或其派生品以及其他独立和无关文档或作品编撰在一个储存卷中或分发媒体上，这称为文档的“聚合体”，前提是编撰成品的著作权对其使用者的法律权限的限制未超出各个独立作品的许可范围。当基于此许可证发布的文档包含在一个聚合体中时，此许可证不适用于聚合体中的本非该文档派生作品的其他作品。

如果第 3 节中的封页文本要求适用于这些文档的副本，则若文档在聚合体中所占的比重小于全作品的一半，文档的封页文本可以放置在聚合体内包含文档部分的封页上，或是电子文档中的等效部分。否则，它必须位于整个聚合体的印刷的封页上。

8. 翻译

翻译被视为一种修改，因此您可以根据第 4 节的条款分发文档的翻译。将固定章节替换为翻译内容需要经得其版权所有者的特别许可，但除了这些固定章节的原始版本之外，您还可以包含一部分或所有固定章节的翻译。您可以包含一个此许可证以及所有许可证声明和免责声明的翻译版本，前提是同时包含它们的原始英文版本。当翻译版本和英文版发生冲突的时候，原始版本有效。

如果在文档中有命名为“致谢”、“题献”或“历史”的章节，保持标题（第 1 节）的要求（第 4 节）恰恰需要更换实际的标题。

9. 终止

除非此许可证中有明确规定，否则您不能对该文档进行复制、修改、分授许可或分发。在此许可证规定外对该文档所进行的任何复制、修改、分授许可或分发都是无效的，并且将自动终止您在此许可证下所拥有的权利。但是，对于在此许可证的规定下从您这里获得副本或权利的各方，只要其完全遵守此许可证的规定，其许可证将不会被终止。

10. 本许可的未来修订版本

自由软件基金会有时会发布 GNU 自由文档许可证的新的修订版版本。这些新版本的主旨和精神与当前版本是一致的，但在解决新问题的具体细节方面可能有所不同。请参见 https://www.gnu.org/copyleft/。

许可证的每个版本都有一个不同的版本号。如果文档指定了适用于它的此许可证“或任何后续版本”的特定带编号版本，则您可以选择遵从指定版本或自由软件基金会发布的任何随后版本（非草稿）的条款和条件。如果文档没有指定此许可证的版本号，您可以选择自由软件基金会发布的任何许可证版本（非草稿）。

附录：如何针对您的文档使用此许可证

```
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”.
```

如果您有固定章节、封面文本和封底文本，请将“with...Texts”部分替换为：

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

如果有不可变章节而没有封页文本，或这三种内容（不可变章节、封面文本、封底文本）的任何其他组合，请合并这两个备选项以适应您的情况。

如果您的文档包含不一般的程序代码示例，建议同时选择自由软件许可证（如 GNU 通用公共许可证）发布这些示例，以允许它们可以用于自由软件。