项目申请书

项目名称：为NestOS适配RISC-V架构

项目主导师：王经纬

申请人：常旭海

邮箱：mahno@libertarian.dev

1. 项目背景

1. NestOS

NestOS 是一款面向云原生和虚拟化场景的不可变操作系统，通过原子化更新、双根文件系统及容器化镜像定制（NFC）优化容器云底噪，结合专为虚拟化设计的组件（NFV）实现高效资源隔离，并提供集群部署工具（NKD）、内核增强、混部调度（Rubik）及无缝迁移方案（x2NestOS），旨在提升大规模容器/虚拟机工作负载的安全性、资源利用率及运维效率。

2. RISC-V指令集架构

RISC-V是一种基于精简指令集（RISC）原则的**开源、免授权、模块化指令集架构**（ISA），具有高度可扩展性和灵活性，可自由定制以适应从嵌入式设备到超级计算机的多样化硬件场景。

2. 项目详细需求

1. nestos-assembler可构建RISC-V64架构的镜像

目前nestos-assembler已可在x86_64构建服务器上通过qemu-kvm自动化构建同硬件架构的NestOS镜像，现需要其可以基于qemu异构模拟跨架构构建RISC-V64架构镜像（iso安装镜像、raw磁盘镜像、qcow2格式qemu镜像等），异构构建的相关代码目前尚待实现。

2. 测试Ignition/overlay/dual-rootfs等功能

需要使用QEMU RISC-V64异构模拟并测试以上功能的可用性，使其最终在RISC-V64架构上可用。

3. 编写CI/CD流水线配置示例

给出NestOS RISC-V自动化构建、部署、测试、运行示例，便于用户根据示例自行构建完整的CI/CD流水线（GitHub Actions或GitLab CI）。

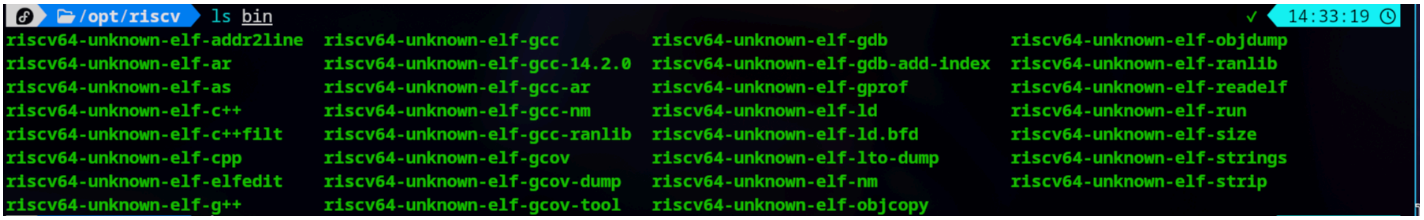
4. 补充用户指南

在文档第 7、8 章新增 RISC-V 构建与部署内容。

3. 技术方法和可行性

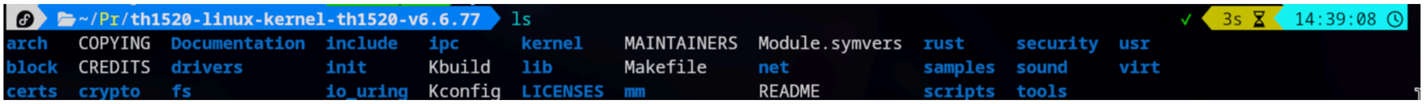
1. Linux内核交叉编译

- 源码构建或从包管理器中获取RISC-V64 GCC编译工具链



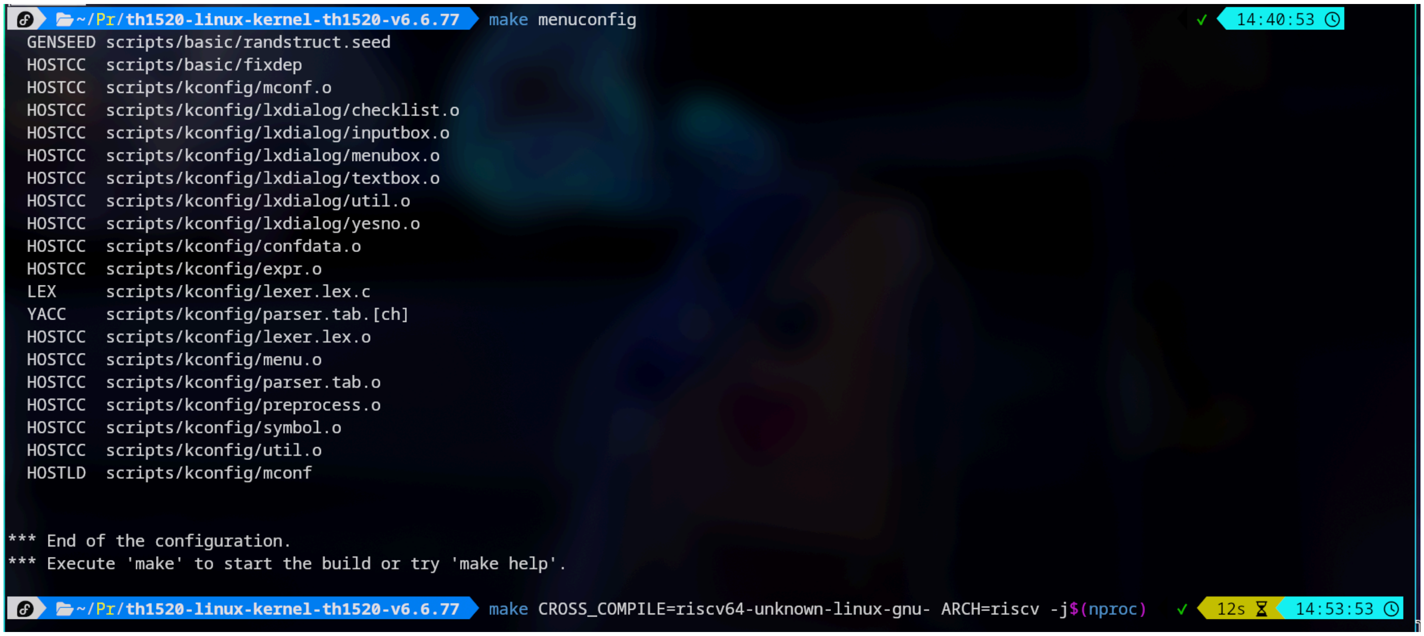
```
 /opt/riscv  ls bin                                                                                    ✓  14:33:19
riscv64-unknown-elf-addr2line  riscv64-unknown-elf-gcc         riscv64-unknown-elf-gdb            riscv64-unknown-elf-objdump
riscv64-unknown-elf-ar         riscv64-unknown-elf-gcc-14.2.0  riscv64-unknown-elf-gdb-add-index  riscv64-unknown-elf-ranlib
riscv64-unknown-elf-as         riscv64-unknown-elf-gcc-ar      riscv64-unknown-elf-gprof          riscv64-unknown-elf-readelf
riscv64-unknown-elf-c++        riscv64-unknown-elf-gcc-nm      riscv64-unknown-elf-ld             riscv64-unknown-elf-run
riscv64-unknown-elf-c++filt    riscv64-unknown-elf-gcc-ranlib  riscv64-unknown-elf-ld.bfd         riscv64-unknown-elf-size
riscv64-unknown-elf-cpp        riscv64-unknown-elf-gcov        riscv64-unknown-elf-lto-dump       riscv64-unknown-elf-strings
riscv64-unknown-elf-elfedit    riscv64-unknown-elf-gcov-dump   riscv64-unknown-elf-nm             riscv64-unknown-elf-strip
riscv64-unknown-elf-g++        riscv64-unknown-elf-gcov-tool   riscv64-unknown-elf-objcopy
```

- 获取指定版本的Linux内核源码



```
 ~/Pr/th1520-linux-kernel-th1520-v6.6.77  ls                                                   ✓  3s ⌛  14:39:08 
arch   COPYING  Documentation  include   ipc      kernel    MAINTAINERS  Module.symvers  rust     security  usr
block  CREDITS  drivers        init      Kbuild   lib       Makefile     net             samples  sound     virt
certs  crypto   fs             io_uring  Kconfig  LICENSES  mm           README          scripts  tools
```

- 通过menuconfig设置所需的功能，并设置环境变量使用指定的编译工具链。



```
 ~/Pr/th1520-linux-kernel-th1520-v6.6.77  make menuconfig                                          ✓  14:40:53 
  GENSEED scripts/basic/randstruct.seed
  HOSTCC  scripts/basic/fixdep
  HOSTCC  scripts/kconfig/mconf.o
  HOSTCC  scripts/kconfig/lxdialog/checklist.o
  HOSTCC  scripts/kconfig/lxdialog/inputbox.o
  HOSTCC  scripts/kconfig/lxdialog/menubox.o
  HOSTCC  scripts/kconfig/lxdialog/textbox.o
  HOSTCC  scripts/kconfig/lxdialog/util.o
  HOSTCC  scripts/kconfig/lxdialog/yesno.o
  HOSTCC  scripts/kconfig/confdata.o
  HOSTCC  scripts/kconfig/expr.o
  LEX     scripts/kconfig/lexer.lex.c
  YACC    scripts/kconfig/parser.tab.[ch]
  HOSTCC  scripts/kconfig/lexer.lex.o
  HOSTCC  scripts/kconfig/menu.o
  HOSTCC  scripts/kconfig/parser.tab.o
  HOSTCC  scripts/kconfig/preprocess.o
  HOSTCC  scripts/kconfig/symbol.o
  HOSTCC  scripts/kconfig/util.o
  HOSTLD  scripts/kconfig/mconf


*** End of the configuration.
*** Execute 'make' to start the build or try 'make help'.

 ~/Pr/th1520-linux-kernel-th1520-v6.6.77  make CROSS_COMPILE=riscv64-unknown-linux-gnu- ARCH=riscv -j$(nproc)  ✓  12s ⌛  14:53:53 
```

2. 使用Butane、Ignition、ostree自动配置操作系统和根文件系统

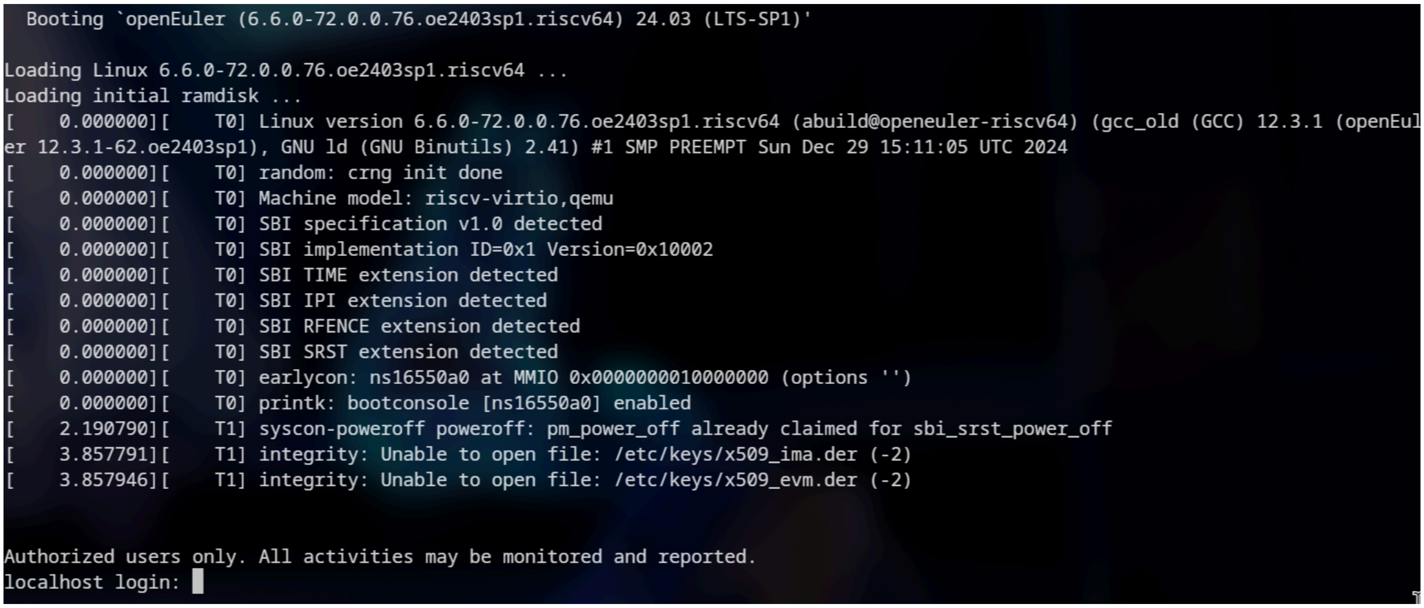
- 使用Butane生成Ignition配置
- 使用Ignition配置自动化首次启动初始化
- 使用ostree提供类似于git的系统原子化更新/回滚机制

3. 理解并修改nestos-assembler源码

- 阅读并理解源码结构，找到新功能代码应添加的位置。仓库地址
- 编写对应的代码

4. qemu虚拟化测试

- 编写启动脚本，测试能否启动qemu虚拟机



```
  Booting `openEuler (6.6.0-72.0.0.76.oe2403sp1.riscv64) 24.03 (LTS-SP1)'

Loading Linux 6.6.0-72.0.0.76.oe2403sp1.riscv64 ...
Loading initial ramdisk ...
[    0.000000][    T0] Linux version 6.6.0-72.0.0.76.oe2403sp1.riscv64 (abuild@openeuler-riscv64) (gcc_old (GCC) 12.3.1 (openEul
er 12.3.1-62.oe2403sp1), GNU ld (GNU Binutils) 2.41) #1 SMP PREEMPT Sun Dec 29 15:11:05 UTC 2024
[    0.000000][    T0] random: crng init done
[    0.000000][    T0] Machine model: riscv-virtio,qemu
[    0.000000][    T0] SBI specification v1.0 detected
[    0.000000][    T0] SBI implementation ID=0x1 Version=0x10002
[    0.000000][    T0] SBI TIME extension detected
[    0.000000][    T0] SBI IPI extension detected
[    0.000000][    T0] SBI RFENCE extension detected
[    0.000000][    T0] SBI SRST extension detected
[    0.000000][    T0] earlycon: ns16550a0 at MMIO 0x0000000010000000 (options '')
[    0.000000][    T0] printk: bootconsole [ns16550a0] enabled
[    2.190790][    T1] syscon-poweroff poweroff: pm_power_off already claimed for sbi_srst_power_off
[    3.857791][    T1] integrity: Unable to open file: /etc/keys/x509_ima.der (-2)
[    3.857946][    T1] integrity: Unable to open file: /etc/keys/x509_evm.der (-2)


Authorized users only. All activities may be monitored and reported.
localhost login: 
```

- 编写自动化测试脚本（python等）

  示例:

```python
import unittest
import subprocess
import select
import time
import sys
import re
try:
    from colorama import init, Fore
    init()
    BLUE = Fore.BLUE
    RESET = Fore.RESET
except ImportError:
    BLUE, RESET = '\033[94m', '\033[0m'
ansi_escape = re.compile(r'\x1b\[([0-9;]*)?[A-Za-z]')

class TestQEMUBootToLoginPrompt(unittest.TestCase):
    TIMEOUT = 60
    LOGIN_PROMPT = "Authorized users only. All activities may be monitored and reported."
    START_SCRIPT = "./start_vm_penglai.sh"
    def setUp(self):
        self.proc = subprocess.Popen(
            [self.START_SCRIPT],
            stdout=subprocess.PIPE,
            stderr=subprocess.STDOUT,
            stdin=subprocess.DEVNULL,
            text=True,
            bufsize=1,
            universal_newlines=True
        )
        self.start_time = time.time()
    def colored_output(self,line):
        timestamp = time.time() - self.start_time
        return f"{BLUE}[+{timestamp:.2f}s]{line}{RESET}"
    def test_boot_to_login_prompt(self):
        prompt_found = False
        while (time.time() - self.start_time) < self.TIMEOUT:
            rlist,_,_ = select.select([self.proc.stdout],[],[],0.5)
            if self.proc.stdout in rlist:
                line = self.proc.stdout.readline()
                if line:
                    sys.stdout.write(self.colored_output(ansi_escape.sub('',line)))
                    if self.LOGIN_PROMPT in line:
                        prompt_found = True
                        break
            if self.proc.poll() is not None:
                break
        self.assertTrue(prompt_found,f"未在 {self.TIMEOUT} 秒内检测到登录提示\n")
    def tearDown(self):
        if self.proc.poll() is None:
            self.proc.terminate()
            try:
                self.proc.wait(timeout=5)
            except subprocess.TimeoutExpired:
                self.proc.kill()

if __name__ == "__main__":
    unittest.main()
```

5. CI/CD流水线搭建

- 自动化构建，自动化测试等
  示例:

```
name: NestOS RISC-V CI example
on:
  pull_request:
    branches: [main]
  jobs:
    test:
      name: "test bootable ISO"
      runs-on: ubuntu-latest
      steps:
        - name: Clone repo
          uses: actions/checkout@v2
        - name: Checkout (HEAD)
          run: git checkout HEAD
        - name: build ISO
        - run: nosa init && nosa fetch && nosa buildextend-live --arch=riscv64
        - name: test ISO
        - run: python3 -m unittest test.py -v
```

4. 规划

1. 第一阶段（7.01-8.15）

- [ ] 手动构建NestOS risc-v架构qemu镜像，并测试相关功能
- [ ] 将正确的构建过程合并入nestos-assembler源代码中

2. 第二阶段（8.15-9.30）

- [ ] 提供CI/CD流水线示例
- [ ] 完善第7、8章的相关文档