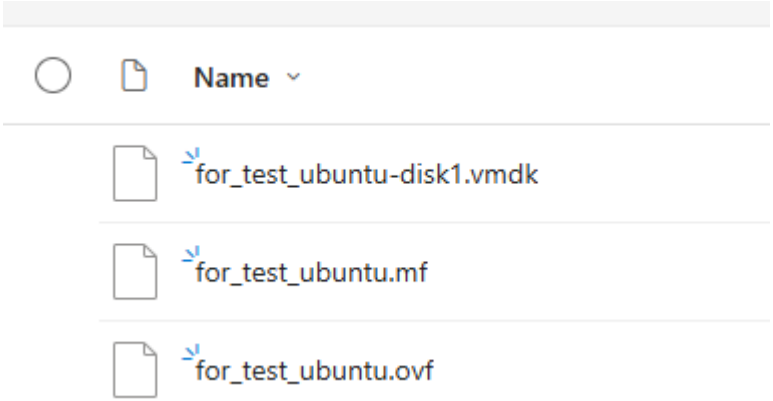
Using Ubuntu VM for Testing

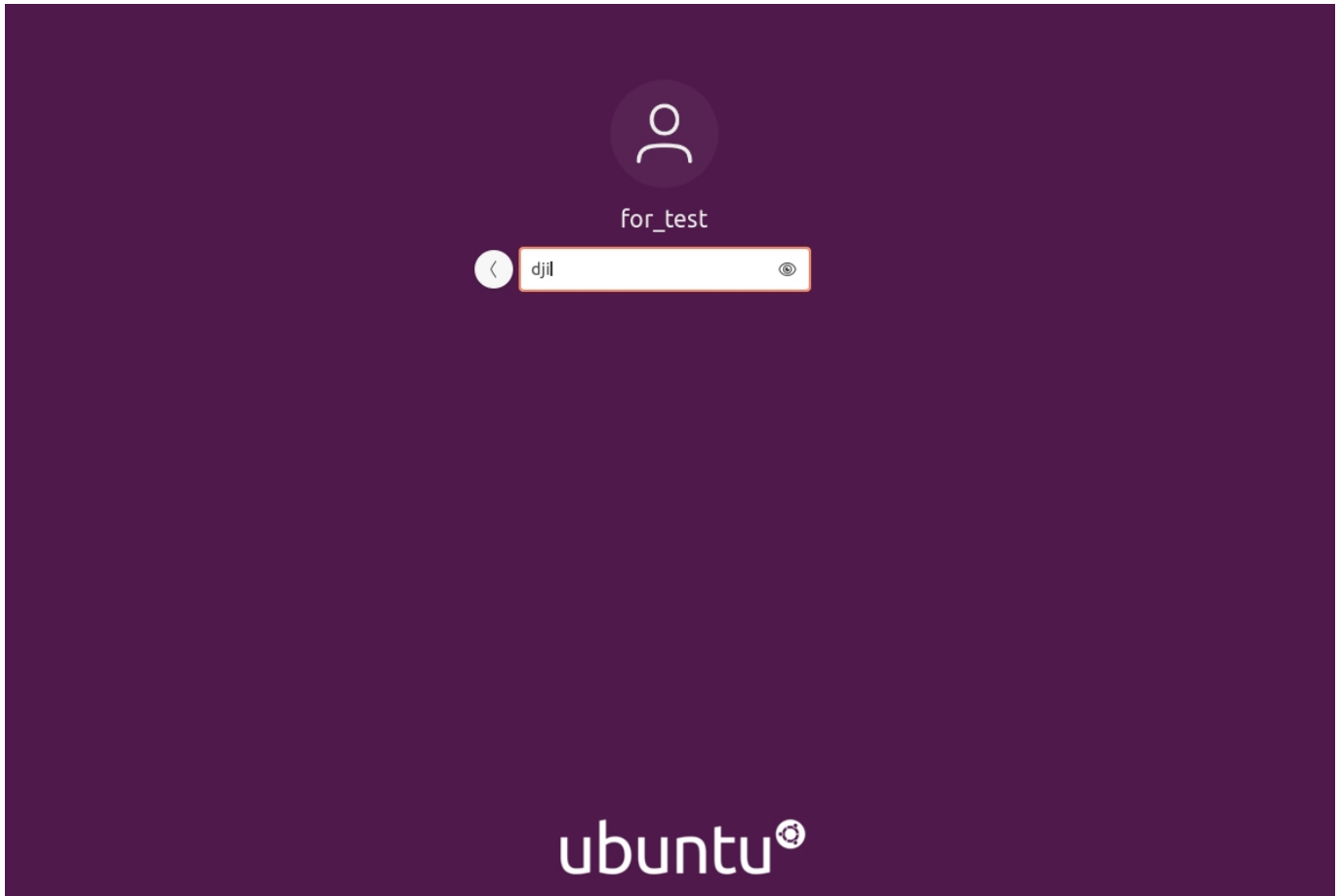
Download Ubuntu .OVF File

- Download link: for_test_VM
- Please download all of the files and put them into one folder.



Open Ubuntu VM

- Please Using Vmware Workstation 16-pro to open the Ubuntu .ovf file.
- If VM is loaded successfully, please login with password: **dji**



Create Dev.img

- Create dev.img for Filesystem mounting before testing.

```
test@ubuntu:~$
test@ubuntu:~$ cd ~/f2fsj/test_dir/
test@ubuntu:~/f2fsj/test_dir$
test@ubuntu:~/f2fsj/test_dir$ ls
test@ubuntu:~/f2fsj/test_dir$
test@ubuntu:~/f2fsj/test_dir$ dd if=/dev/zero of=dev.img bs=1M count=16384
16384+0 records in
16384+0 records out
17179869184 bytes (17 GB, 16 GiB) copied, 108.949 s, 158 MB/s
test@ubuntu:~/f2fsj/test_dir$ ls
dev.img
test@ubuntu:~/f2fsj/test_dir$
test@ubuntu:~/f2fsj/test_dir$
```

Run Setup Script

- Testing environment setting.

```
test@ubuntu:~$ cd f2fsj/filebench/script/
test@ubuntu:~/f2fsj/filebench/script$
test@ubuntu:~/f2fsj/filebench/script$
test@ubuntu:~/f2fsj/filebench/script$ ls
ext4_fb.sh  f2fs_fb.sh  f2fsj.ko  j_f2fs_fb.sh  perf.sh  setup.sh  xfs_fb.sh
test@ubuntu:~/f2fsj/filebench/script$
test@ubuntu:~/f2fsj/filebench/script$ sudo ./setup.sh
[sudo] password for test:
nodev   autofs
nodev   binfmt_misc
->>>>sudo -i to setup randomize_va_space
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
kernel.perf_event_paranoid = -1
vm.dirty_background_ratio = 5
vm.dirty_ratio = 50
vm.dirty_writeback_centisecs = 1500
```

Test Scripts Introduction

- At ~/f2fsj/filebench/script, there are four scripts for f2fsj/f2fs/ext4/xfs testing.

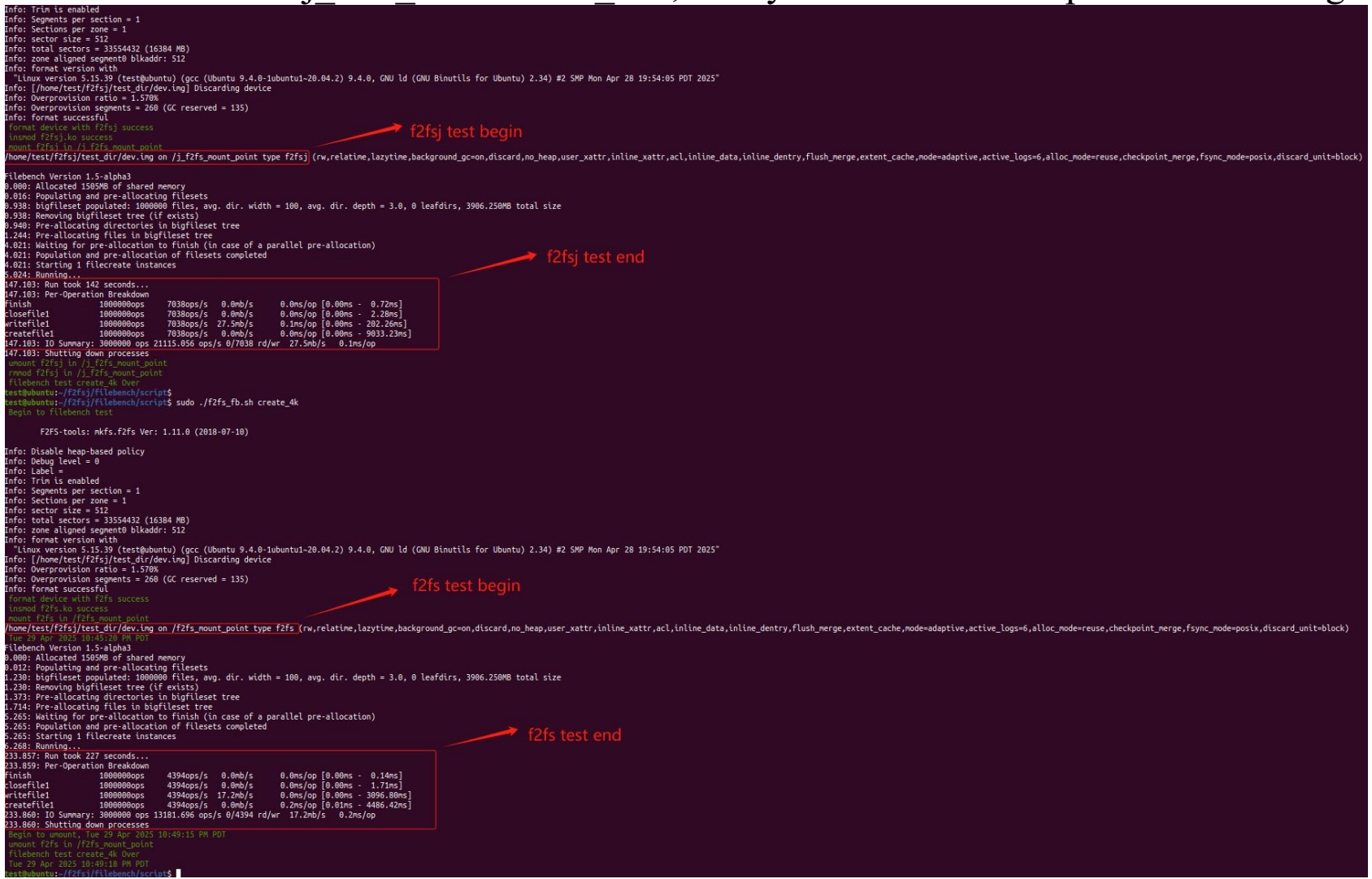
```
test@ubuntu:~/f2fsj/filebench/script$ ls
ext4_fb.sh  f2fs_fb.sh  j_f2fs_fb.sh  perf.sh  setup.sh  xfs_fb.sh
test@ubuntu:~/f2fsj/filebench/script$
test@ubuntu:~/f2fsj/filebench/script$
```

- Benchmarks.
  - We use shell scripts to integrate the benchmarks.
  - Using –h, then you can see the benchmarks for testing.
  - Please note that due to limited size of dev.img, VM does not support big file benchmarks like create_64g/rread_64g/rwrite_64g.

```
test@ubuntu:~/f2fsj/filebench/script$ sudo ./j_f2fs_fb.sh -h
-------------------------------------------
META Only test cases:
./xx_fb.sh create_empty, create empty files
./xx_fb.sh unlink_empty, delete empty files
./xx_fb.sh mkdir,        make directory
./xx_fb.sh rmdir,        remove directory
./xx_fb.sh readdir,      read directory
-------------------------------------------
DATA_And_META test cases:
./xx_fb.sh create_4k,          create files with 4kb size
./xx_fb.sh create_32k,         create files with 32kb size
./xx_fb.sh read_small          read 4M files(4k for each)
./xx_fb.sh copy_4k,            copy files with 4kb size
./xx_fb.sh create_1g,          create files with 1GB size
./xx_fb.sh delete_4k,          delete files with 4kB size
./xx_fb.sh rread_64g           random read files with 64g size
./xx_fb.sh rwrite_64g          random write files with 64g size
./xx_fb.sh seqread_64g         sequence read files with 64g size
./xx_fb.sh seqwrite_64g        sequence write files with 64g size
./xx_fb.sh create_64g          create a file with 64g size
-------------------------------------------
./xx_fb.sh varmail,      realistic workloads for varmail
./xx_fb.sh oltp,         realistic workloads for oltp
./xx_fb.sh fileserver,   realistic workloads for fileserver
./xx_fb.sh webproxy,     realistic workloads for webproxy
./xx_fb.sh webserver,    realistic workloads for webserver
-------------------------------------------
./xx_fb.sh test, just test script
-------------------------------------------
test@ubuntu:~/f2fsj/filebench/script$
```

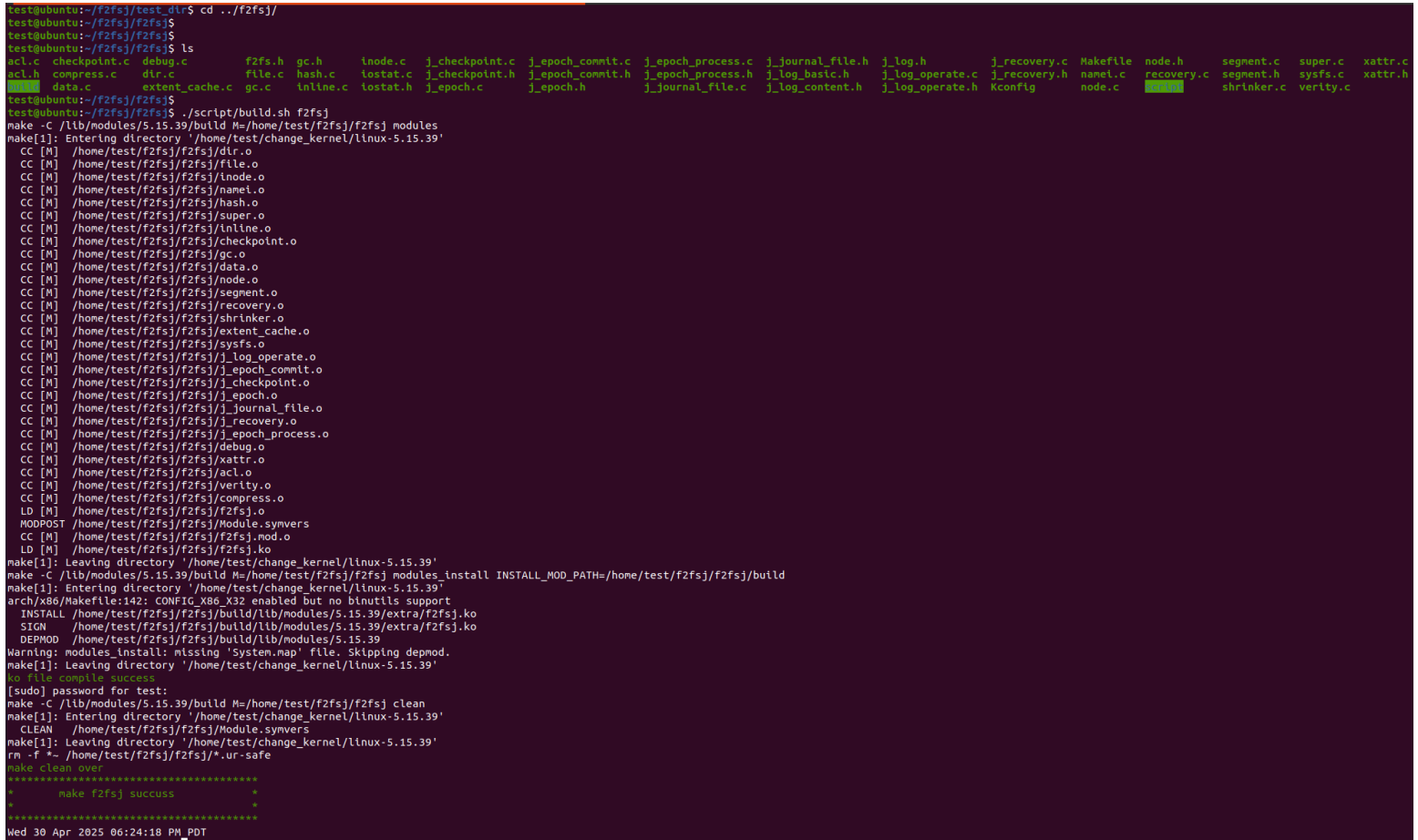
Run Benchmarks

- Run benchmarks like “Sudo ./j_f2fs_fb.sh create_4k”, then you can see the output like following figure.



Code Compile

- Also, we provide compile script to build f2fsj.
- Please goto ~/f2fsj/f2fsj; Then using “./script/build.sh f2fsj” to build.



Problems that may be encountered

- Please using Vmware Workstation 16 pro to load VM because Other versions may lead to unsuccessful import.
- The hardware configuration of the virtual machine is 8 CPUs and 8GB of memory. You can adjust it according to your actual situation.
- The running results in the screenshot are for reference only, as the results may vary depending on different hardware configurations.
- In order to adapt to the environment described in the article, we have replaced the kernel of Ubuntu 20.04.3 with version 5.15.39, which is not from the Ubuntu distribution. Considering the uncertainties in the environment transplantation, if you encounter any problems during use, please feel free to contact us!