技术选择：

本项目选择了 bpftrace 来进行数据采集。bpftrace 是一种高级跟踪语言，提供了丰富的跟踪指令和语法，使跟踪和分析更加灵活和高效。 bpftrace 基于 Linux 的 eBPF 技术，通过探针机制采集内核和程序运行的信息，具有轻量级和低性能开销的优点。

算法预测部分则采用 python 语言编写。

项目结构：

本项目分为两部分，分别为数据采集部分与结果预测部分。

其中数据采集部分使用 bpftrace 编写(runqlen.bt/oepnsnoop.bt/biosnoop.bt)。

结果预测部分由 python 部分编写，其中 runqlen.py 使用逻辑回归进行预测，opensnoop.py 使用了向量化以及随机森林的方法。

使用说明：

本项目利用 bpftrace 采集容器运行时的数据。具体使用方法为：

1. 打开一个已安装好 bpftrace 的容器：

```
cooler@ubuntu:~/Desktop/bpf_tools$ sudo docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS         PORTS     NAMES
0a15fb516985   ebpf-for-mac   "/bin/sh -c 'mount -…"   43 hours ago   Up 23 hours              ebpf-for-mac
cooler@ubuntu:~/Desktop/bpf_tools$ sudo docker exec -it 0a15fb516985 /bin/bash
```

其中，ebpf-for-mac 是已安装好 bpftrace 的容器，启动该容器，进入到含有 bpftrace 程序运行脚本的文件夹中。也可和主机共享文件夹，通过 docker run-v 在容器中设置一个挂载点，并将主机上含有 bpftrace 的文件目录关联到该挂载点下。例如运行以下命令：

```
docker run -itd \
  --name ebpf-for-mac \
  --privileged \
  -v /lib/modules:/lib/modules:ro \
  -v /etc/localtime:/etc/localtime:ro \
  -v /home/cooler/Desktop/bpf_tools:/root/bpf_tools \
  --pid=host \
  ebpf-for-mac
```

2. 运行执行脚本 run_bpftrace.sh

```
root@0a15fb516985:~/bpf_tools# ./run_bpftrace.sh
bpftrace has been terminated and output saved to bt.log
```

3. 输出日志，使用 python 利用容器输出的日志文件 bt.log，在主机中进行预测。

判定 CPU 异常行为的指标为 runqlen，相应文件为 runqlen.py

```
cooler@ubuntu:~/Desktop/bpf_tools$ python3 runqlen.py
Model accuracy on training set: 0.94
Predictions for new samples: [0]
```

```
Model accuracy on training set: 0.94
Predictions for new samples: [1]
```

其中 0 为正常，1 为异常。

判定 C 磁盘异常行为的指标为 opensnoop 与 biosnoop。由于 dataset 中 biosnoop 原始数据为七列，bpftrace 输出 log 中数据为五列，无法进行精准预测，故只采用 opensnoop 一个指标，相应文件为 opensnoop.py

```
cooler@ubuntu:~/Desktop/bpf_tools$ python3 opensnoop.py
Model Accuracy: 92.31%
Predicted Behavior for Example Process: Normal
```

其中 normal 为正常，abnormal 为异常。

可改进的地方：

使用容器以及主机的 cron 服务来进行定时检测。

代码文件说明：

1.runqlen.bt

主要功能是监控和统计当前系统的 CFS 运行队列长度。代码说明如下：

（1）初始化提示：在脚本开始时，打印一条消息告知用户采样频率和终止方式。

（2）定时采样：以 50Hz 的频率定期采样。

（3）获取运行队列长度：获取当前任务的 task_struct->从 task_struct 中提取所属的 CFS 运行队列->获取运行队列中正在运行的任务数量，并减去自身任务，得到实际等待运行的任务数。

（4）记录数据：将运行队列长度记录到直方图 @runqlen 中，以便后续分析和可视化。

log 文件输出格式：

```
Attaching 2 probes...
Sampling run queue length at 99 Hertz... Hit Ctrl-C to end.


@runqlen:
[0, 1)            1918 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@|
[1, 2)               2 |                                                    |
```

2.oepnsnoop.bt

主要功能是追踪系统调用 open 和 openat 的执行情况，并输出每次调用的详细信息，包括进程 ID、进程名、文件描述符、错误码和打开的文件路径。代码说明如下：

（1）打印提示信息和表头，用于显示追踪结果。

（2）追踪 open 和 openat 系统调用进入事件。

（3）追踪 open 和 openat 系统调用退出事件。

（4）清除关联数组 @filename，释放资源。

log 文件输出格式：

```
Attaching 6 probes...
Tracing open syscalls... Hit Ctrl-C to end.
PID    COMM               FD ERR PATH
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-co
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-co
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-co
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-co
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-co
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-co
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-co
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-co
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-co
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-co
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-co
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-co
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-co
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-sc
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-sc
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/kernel/sound/pci/hda/snd-hda-sc
7630   depmod              2   0 /lib/modules/5.15.0-130-generic/modules.builtin.bin
```

3.biosnoop.bt

主要功能是追踪块设备的 I/O 操作，记录每个 I/O 请求的开始时间和结束时间，并计算其延迟（以毫秒为单位）。

代码说明如下：

（1）打印表头，用于显示追踪结果的列标题。

（2）追踪开始：当 I/O 操作开始时，记录当前时间、进程 ID、进程名和相关磁盘名称。

（3）追踪结束：当 I/O 操作结束时，计算延迟，打印详细信息，并清理记录的数据。

（4）结束清理：在脚本结束时，清除所有关联数组。

log 文件输出格式：

```
Attaching 6 probes...
cannot attach kprobe, probe entry may not exist
Warning: could not attach probe kprobe:__blk_account_io_done, skipping.
cannot attach kprobe, probe entry may not exist
Warning: could not attach probe kprobe:__blk_account_io_start, skipping.
TIME(ms)     COMM            DISK    PID    LAT(ms)
428          NetworkManager  sda     794          2
429          NetworkManager  sda     794          0
430          jbd2/sda5-8     sda     343          1
431          jbd2/sda5-8     sda     343          1
431          jbd2/sda5-8     sda     343          0
958          kworker/u256:3  sr0     5562         0
1858         snapd           sda     809         13
1860         snapd           sda     809          1
1860         snapd           sda     809          0
1861         snapd           sda     809          1
1862         snapd           sda     809          0
1864         snapd           sda     809          1
1864         snapd           sda     809          1
1864         snapd           sda     809          2
1876         snapd           sda     809          0
2973         kworker/u256:0  sr0     5673         0
4000         snapd           sda     809         14
4002         snapd           sda     809          1
4002         snapd           sda     809          1
4002         snapd           sda     809          1
4002         snapd           sda     809          0
4003         snapd           sda     809          0
4003         snapd           sda     809          0
4484         snapd           sda     809          0
4484         jbd2/sda5-8     sda     343          0
4485         jbd2/sda5-8     sda     343          0
4486         jbd2/sda5-8     sda     343          0
4486         jbd2/sda5-8     sda     343          0
4607         kworker/u256:0  sda     5673         1
4607         kworker/u256:0  sda     5673         1
4607         kworker/u256:0  sda     5673         1
4607         kworker/u256:0  sda     5673         1
4607         kworker/u256:0  sda     5673         1
4607         kworker/u256:0  sda     5673         1
4607         kworker/u256:0  sda     5673         1
4607         kworker/u256:0  sda     5673         1
4608         kworker/u256:0  sda     5673         1
```

4.run_bpftrace

代码的自动运行脚本脚本，可定义脚本路径，实现框架的灵活拓展。

5.runqlen.py

读取 log 文件，并使用逻辑回归对数据进行预测。该文件读取的数据可以归结为一个二分类问题，因此采用逻辑回归算法检测数据是否异常。逻辑回归是一个线性模型，容易理解和实现。并且该算法对于较小的特征空间和线性可分的数据集，训练和预测速度较快。代码通过 sklearn 使用了这一算法。输出结果为 0/1，其中 0 为正常，1 为异常。

6.opensnoop.py

读取 log 文件，将数据向量化，并使用随机森林对数据进行预测。该文件读取的数据有多个特征，随机森林算法可以处理具有很多特征的数据的数据，并且不用降维，无需做特征选择，判断特征的重要程度，判断出不同特征之间的相互影响，并且实现起来比较简单。代码通过 sklearn 使用了这一算法。输出结果为 normal/abnormal，其中 normal 为正常，abnormal 为异常。