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。