系统帮助说明

设计

参见设计文档。

环境

OS

```
➜  ~ uname -a
Linux ubuntu 3.13.0-32-generic #57-Ubuntu SMP Tue Jul 15 03:51:08 UTC 2014 x86_
```

SDN

1. Open vSwitch

```
➜  ~ sudo ovs-vsctl --version
ovs-vsctl (Open vSwitch) 2.3.2
Compiled Feb 25 2016 00:59:19
DB Schema 7.6.2
```

2. SDN Controller： POX

测试数据集

- Linux kernel的100个版本
- redis server的37个版本

系统主要功能

- 文件上传
- 网络中SDN控制器重复数据检测
- 服务器端存储文件，索引构建

运行

- 启动OVS，（start_ovs.sh）

```
➜  openvswitch-2.3.2 cat start_ovs.sh
sudo make
sudo make install
sudo make modules_install
sudo /sbin/modprobe openvswitch
/sbin/lsmod | grep "openvswitch"
sudo mkdir -p /usr/local/etc/openvswitch
sudo ovsdb-tool create /usr/local/etc/openvswitch/conf.db vswitchd/vswitch.ovss
sudo ovsdb-server --remote=punix:/usr/local/var/run/openvswitch/db.sock \
                     --remote=db:Open_vSwitch,Open_vSwitch,manager_options \
                     --private-key=db:Open_vSwitch,SSL,private_key \
                     --certificate=db:Open_vSwitch,SSL,certificate \
                     --bootstrap-ca-cert=db:Open_vSwitch,SSL,ca_cert \
                     --pidfile --detach
sudo  ovs-vsctl --no-wait init
sudo ovs-vswitchd --pidfile --detach
```

- 启动redis server， POX controller需要，缓存指纹

```
cd redis-3.0.5/     # before sdn controller start redis server

./src/redis-server # if change maxmemory config, then ./redis-server redis.conf
```

- 启动POX，指定我们实现的模块 pox.dedu.dedupe06

```
cd pox/         # start sdn controller

./pox.py log.level --DEBUG pox.dedu.dedupe06
```

- 启动Mininet

```
cd mininet/
sh start_mn.sh

iperf h1 h10
```

- 修改流表

```
./init_flow_15_switches.sh    # add flows to switch use bash !!
```

- 编译client，server

```
cd source-dedu/   # compile
sh make10.sh
```

- 在Mininet的两个host中分别运行client，server

```
h15 ./server10  &  # start backup server
h1 ./client09  h15 <dataset>    # client backup a version
```

- 备份完成后，统计信息

```
➜  SdnBasedDeduplication : cd test
➜  test : python extract_log.py
Enter file name: kernel_sdna_cache_800k
--------------------------dedu time-------------------
[4.008264, 3.943874, 4.339101, 3.975689, 3.926724, 3.859574, 4.778007, 4.488389
--------------------------file count------------------
[561, 598, 603, 608, 620, 637, 667, 671, 676, 688, 690, 707, 711, 729, 740, 766
---------------------------file new count-------------
[0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 2, 1, 0, 0, 0, 0, 1,
----------------------------file new count from pox---
[561, 319, 202, 96, 102, 63, 328, 187, 146, 142, 16, 139, 140, 129, 97, 96, 233
----------------------------average dedu ratio--------
0.852354046734
```

参数配置

配置client/server的去重方法

```
$ ./client <server_ip_addr> [base/bloom/sdna] <backup_dir>
$ ./server -m [bloom/sdna]
```

Bloom Filter大小

需要在程序中修改，server, SDN Controller

Cache大小

修改SDN Controller使用的redis server的内存配置(redis.conf)，maxmemory以及替换策略

```
maxmemory 1MB
maxmemory-policy allkeys-lru
```

带宽大小

在启动Mininet时候配置, bw带宽，delay时延

```
sudo mn --topo=linear,20 --link tc,bw=1000,delay=1ms --mac --switch=ovsk --cont
```

网络规模

在启动Mininet时候配置,配置OVS个数

```
sudo mn --topo=linear,40 --link tc,bw=1000,delay=1ms --mac --switch=ovsk --cont
```

性能测试结果

- 三种去重方法备份时间

```
bloom_time = (35.765523, 26.453468, 22.883979, 21.731732, 22.489979, 21.537492,

base_time = (36.472232, 25.987929, 23.078874, 20.864176, 20.846923, 20.270137,
sdna_time  = [3.52194, 3.326431, 3.422871, 3.630502, 3.222517, 3.230395, 4.2105
```

- Bloom filter 大小影响备份时间
- cache大小影响备份时间
- 带宽大小影响备份时间
- 网络规模影响备份时间
- 重删率
- 开销（查看流表的matches个数）

注：绘图使用matplotlib

性能分析

- 与传统源端去重相比，sdna明显减小备份时间

- Bloom filter过小，发生false positive的概率升高，使得SDN控制器重复数据检测效果不明显
- cache太小同样会使得备份时间增加
- 在低带宽环境下，sdna备份时间的减小更加明显
- 重删率有稍微的下降，由于Bloom filter的false negative

运行截图