CernVM-FS Profiling

Razvan-Nicolae Virtan

Supervisors: Radu Popescu, Jakob Blomer

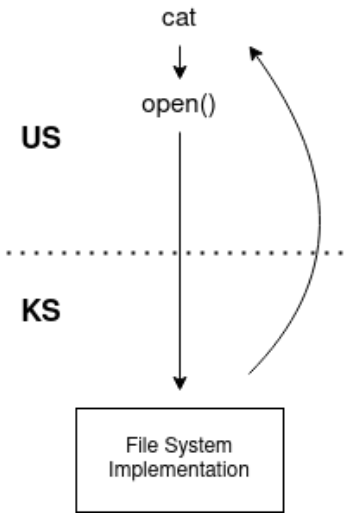
About CernVM-FS *(CVMFS)*

- created and optimized to deliver scientific software stacks to a distributed compute grid
- offers a file system interface for software repositories

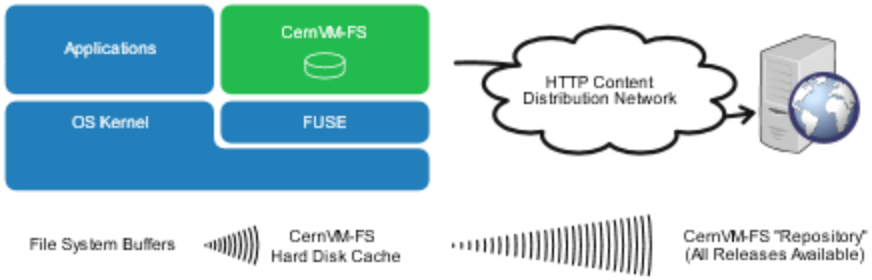
[razvan@~]$ sudo mount -t cvmfs atlas.cern.ch /cvmfs_mount/atlas.cern.ch/ # **new cvmfs process created on local system**

[razvan@~]$ cat /cvmfs_mount/atlas.cern.ch/repo/test

Classic scenario

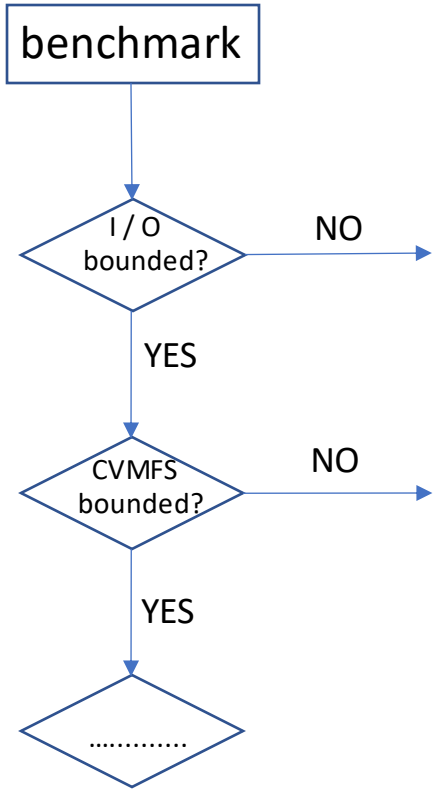


CVMFS

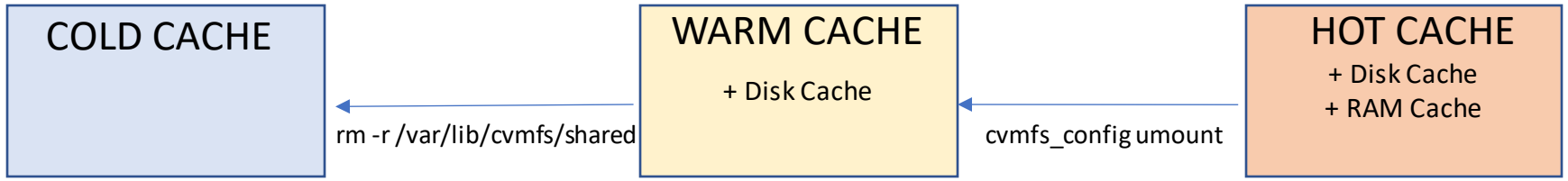


CVMFS Profiling

1. Develop a general set of tools & procedures for analyzing CVMFS performance
2. Apply these tools on some known benchmarks and spot possible bottlenecks



I / O bounded? avg_cache_time



$ ./profiling_tools/avg_cache_time.sh tensorflow_benchmark 2

Cache_Type	Real Avg	User Avg	Sys Avg
cold	25.560	10.639	1.282
warm	12.924	10.359	1.183
hot	11.690	10.051	0.959

Cache_Type	CPU %	BLOCKED %
cold	0.399	0.601
warm	0.893	0.107
hot	0.942	0.058

Compare Case	Real Time	User Time	Sys Time
cold / hot	2.192	1.058	1.337
cold / warm	1.992	1.027	1.082
warm / hot	1.106	1.031	1.233

CVMFS bounded? cvmfs_talk

- How much of the blocked time is actually spent in CVMFS?
- Measure time spent in cvmfs callbacks and calculate the total

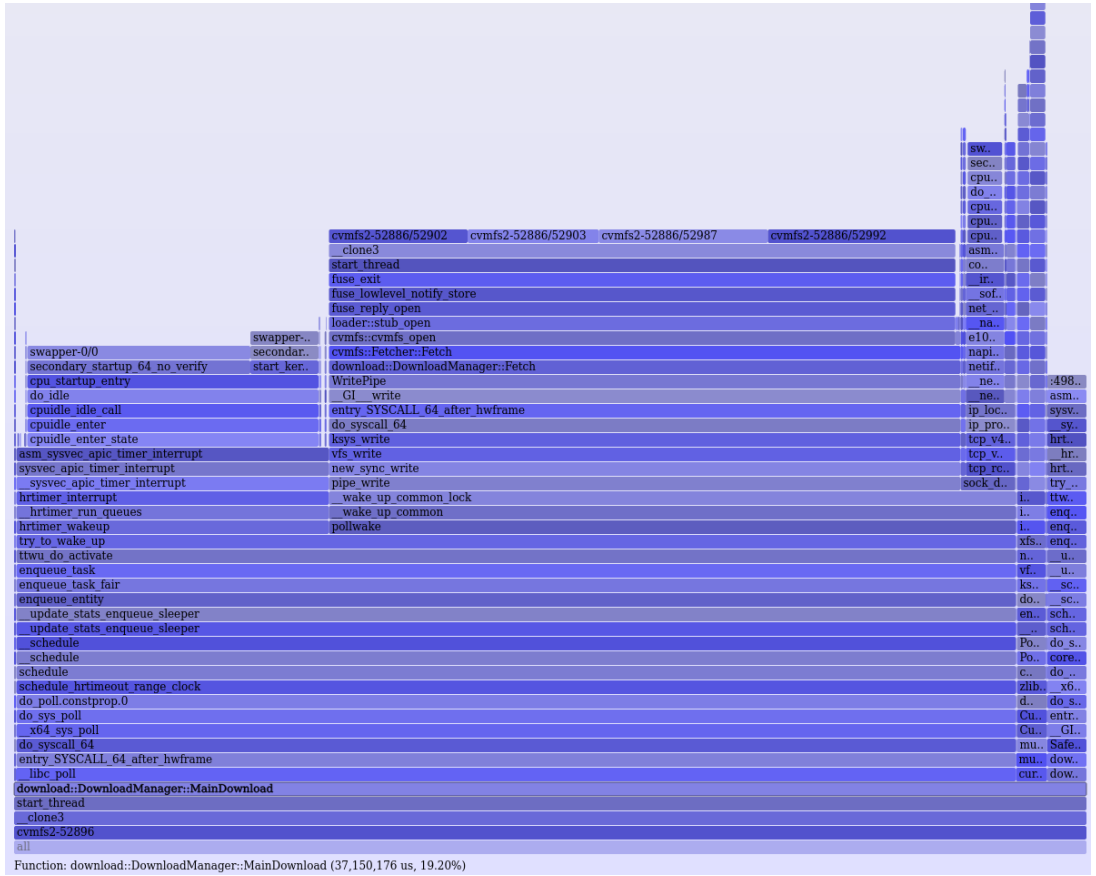
$$T_{Callbacks} / T_{Blocked}$$

```
$ cvmfs_talk -i unpacked.cern.ch internal affairs
....
Total Time In Callbacks = 34945ms
....
```

Where is the bottleneck? generate_flamegraphs.sh

- **Method 1**: exhaustive ON / OFF CPU analysis using flamegraphs

  $ ./profiling_tools/generate_flamegraphs.sh --oncpu --dwarf --benchmark lhcb_benchmark --cache hot

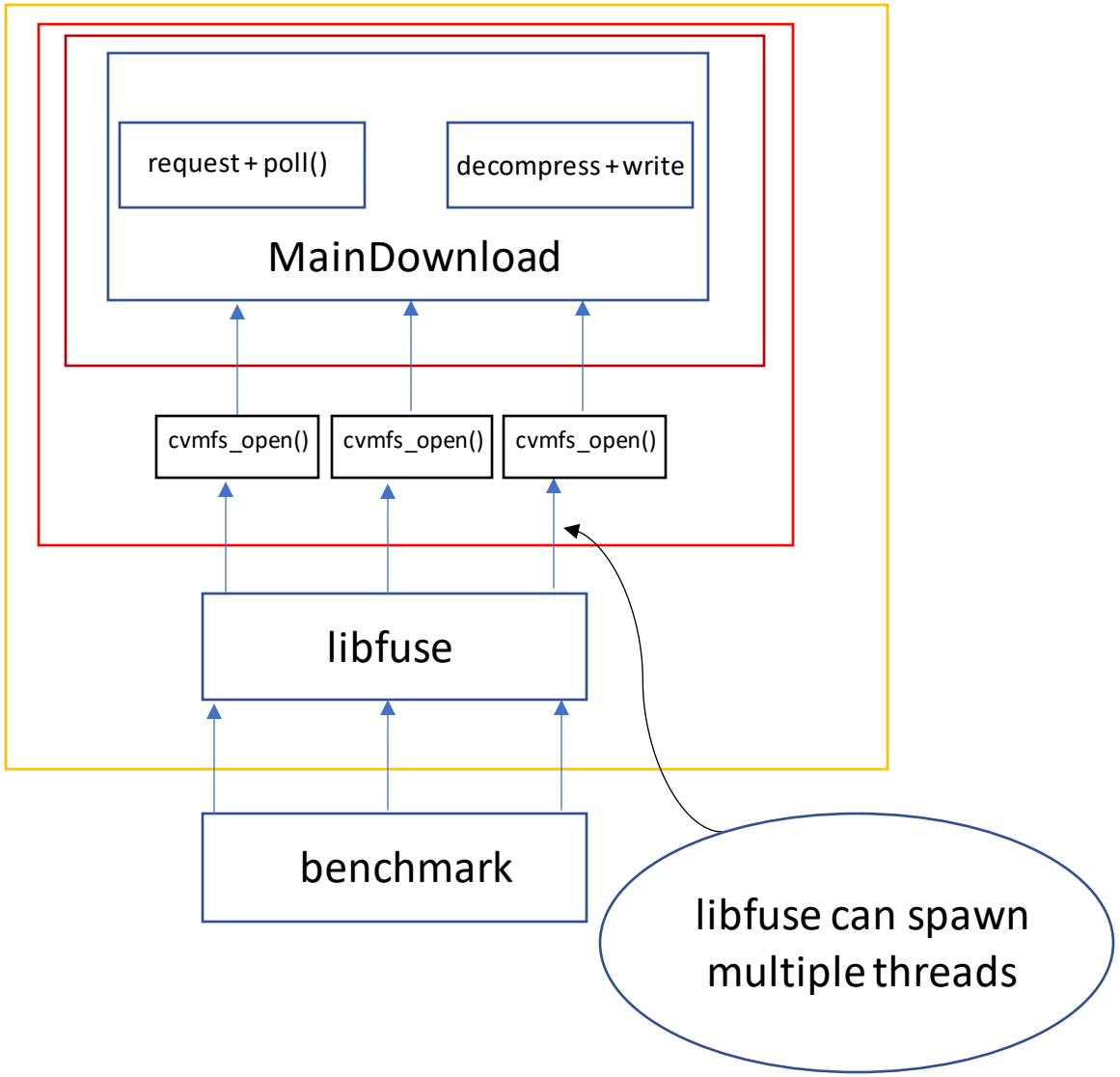


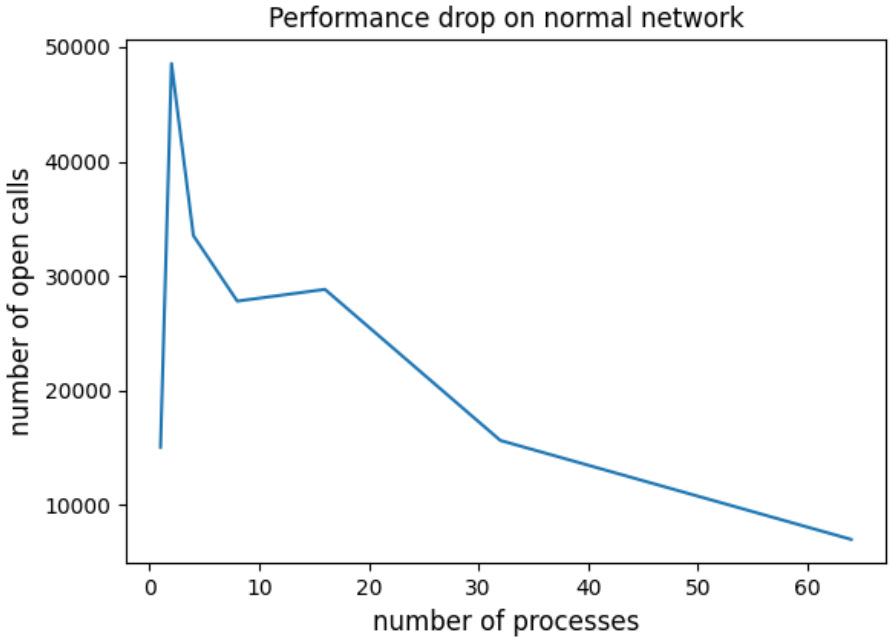
Where is the bottleneck? cvmfs_talk

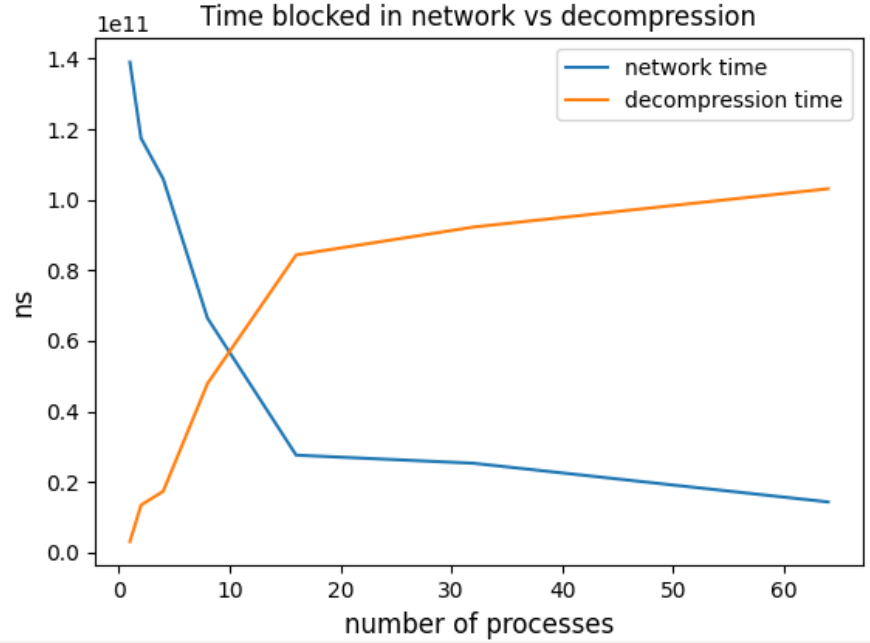
- Flamegraphs are inaccurate / expensive and can fail on multi-threaded scenarios.
- **Method 2**: add more timers and counters in the cvmfs code, in places that can become bottlenecks.

Benchmark scenario:

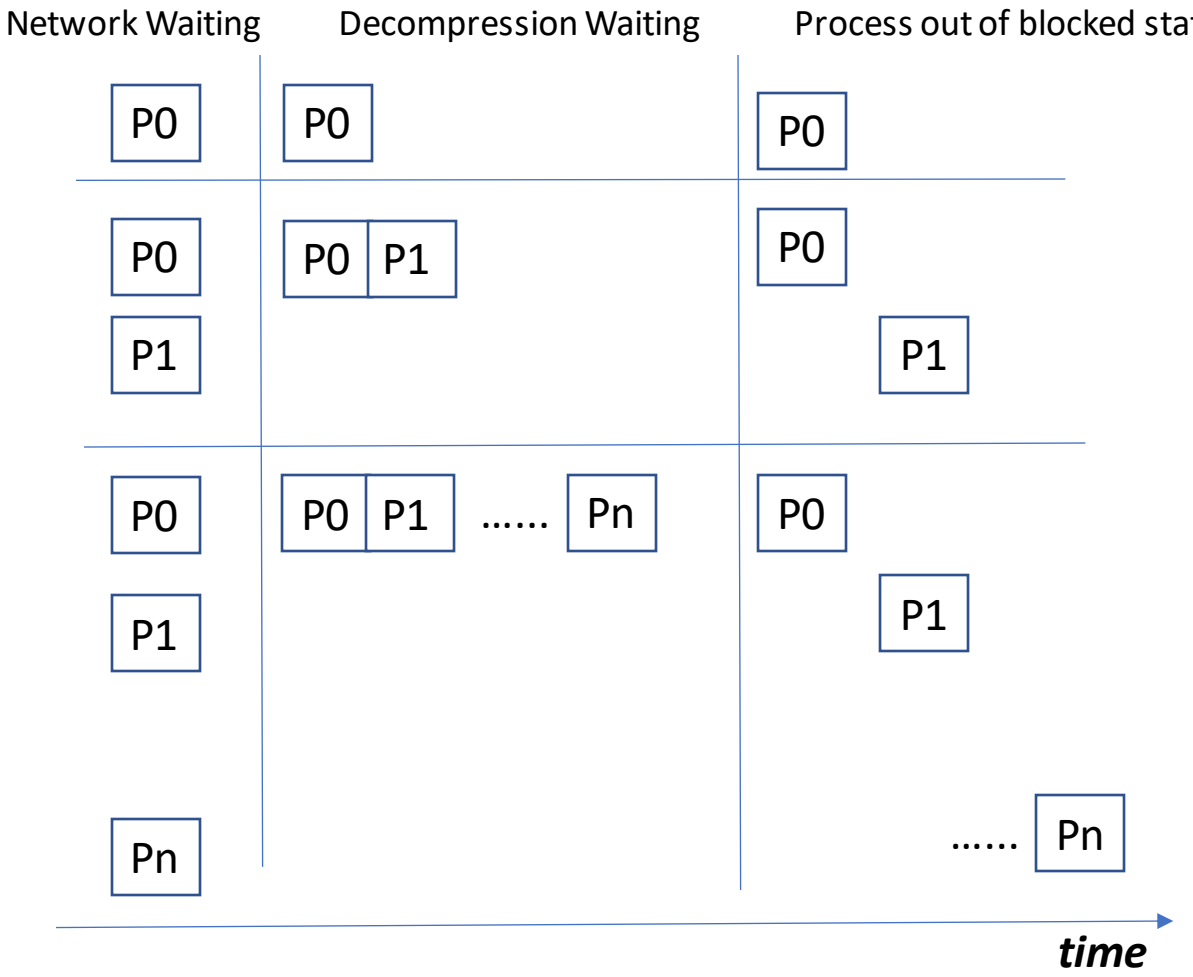
- Multi core system
- Cold Cache
- Multiple processes, attempting to access different data from the same repo

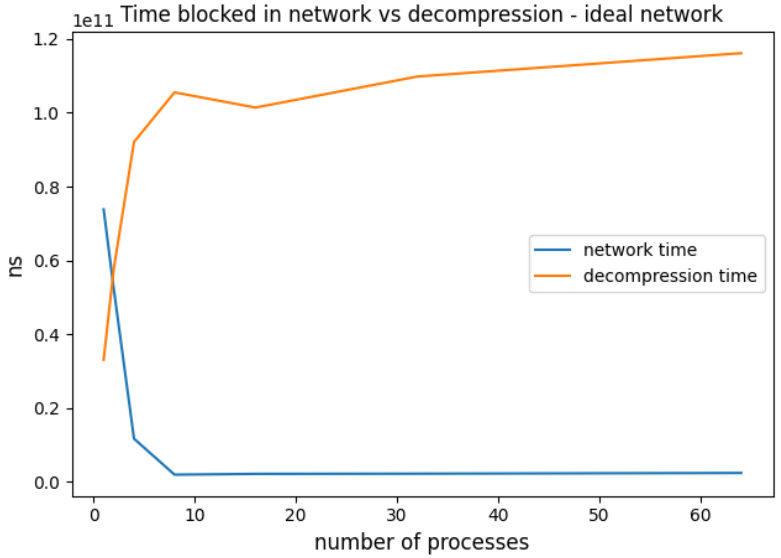

request + poll()
decompress + write
MainDownload
cvmfs_open()
cvmfs_open()
cvmfs_open()
libfuse
benchmark
libfuse can spawn multiple threads

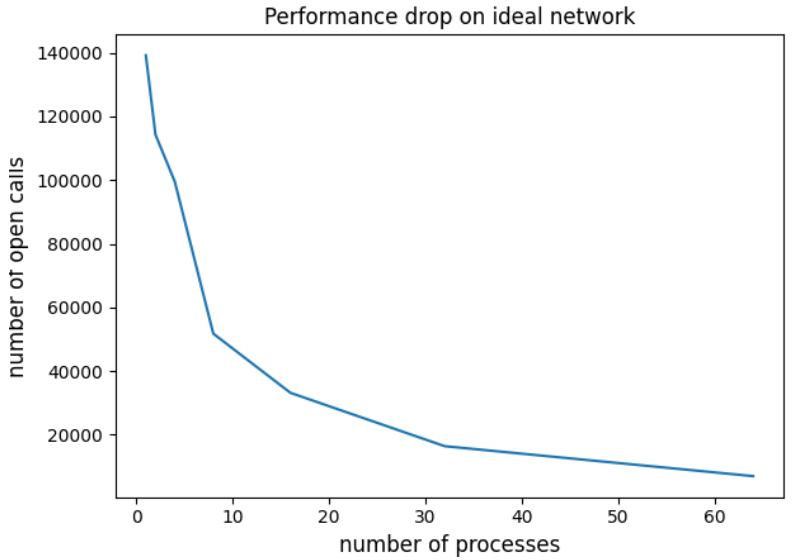

Performance drop on normal network
number of open calls
number of processes
50000
40000
30000
20000
10000
0
10
20
30
40
50
60


Time blocked in network vs decompression
1e11
ns
number of processes
network time
decompression time
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
50
60

The reason behind the bottleneck







Processes queuing for decompression