# QoS-Aware Virtualization (using SSD and disk)

Christopher Feener
Advisor: Xuechen Zhang

WASHINGTON STATE UNIVERSITY VANCOUVER

## Motivations

- SSD has high performance, and is becoming increasingly economical and efficient.
- SSD can be an extension of DRAM.
- Disk is very cheap and durable.

## Challenges

- Speed of SSD is fast, but not durable.
- Disk is durable, but slow.
- (Last attempt of “Hybrid Swapping” was done on Linux kernel 2.6, and many parts of the kernel have since changed.)
- (Lack of documentation.)

## Our Approach

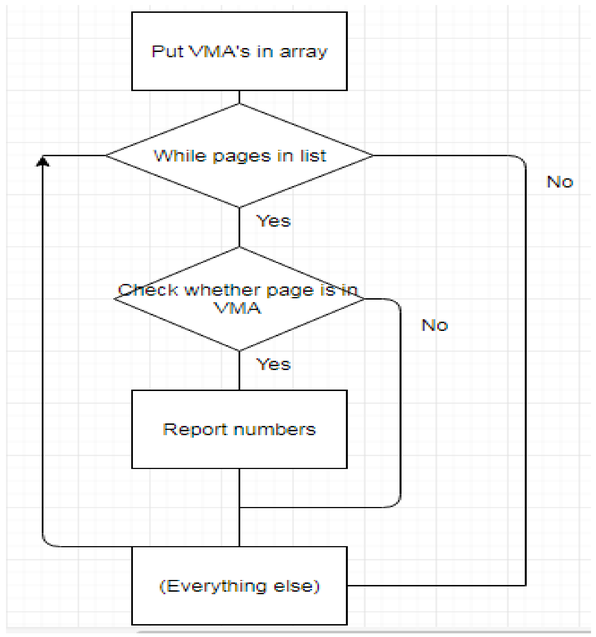
- “Hybrid Swapping” could use the advantages of both speed and limited swapping. It did this by swapping only page faults to disk.
- Only minor additions were made to linux kernel 4.11, specifically in the function *shrink_page_list()*.
- Debugging: No major program/method was used other than dmesg and printk statements.

## Results and Data

- Overall, writes to SSD were reduced by 16 - 40% without hurting performance greatly ( -0.8 – 7.1%).
- Memcached writes were reduced by 37%, though performance decreased slightly by 0.8%.
- ImageMagick had 40% fewer writes, 4.6% more performance. Had the largest reduction of writes.
- Correlation Computation had 22% fewer writes, 7.1% more performance. Had the best improvement in performance.
- Matrix Inverse had 16% fewer writes, 0.5% more performance.
- Using Sequences: Hybrid Swap had 49% fewer page faults than SSD-swap, and 39% fewer than RAID-swap, due to finding page access patterns.
- Memory Size: However, throughput testing in Memcached showed Hybrid Swap to use more throughput than RAID-swap. (22% for 512 MB sizes, and identical for 2 GB)
- Performance: Hybrid Swap's run time was about the same as SSD-swap's and much better than RAID-swap's. The number of major faults in Hybrid Swap was better than both the others'. This is partly due to better disk efficiency.
- Concurrency: Hybrid Swap's performance was better than both SSD-swap's (by 20%) and RAID-swap's (43%).
- Bounds for penalty ratio: Hybrid Swap follows limits on ratio of page fault penalty to runtime, unlike SSD- and RAID-swap.

## Immediate goal: Spatial Locality

- The “Hybrid Swap” method was meant to use both spatial and temporal locality.
- Spatial locality deals with how close addresses from a sequence are.
- By iterating through processes from the current task, virtual memory areas (VMA's) were put in an array.
- As the function iterated through pages, each page was checked to all the VMA's, using *page_address_in_vma()*.
- Each “match” had its information printed.
- These matches could be saved and recorded in another array for future use, such as analyzing the average distance between page addresses.




## Setup

- Started with Qemu program, booting from kernel source code inside.
- Tested with ImageMagick on a large image of a galaxy. This forced the computer to have page faults and use swap.
- Coded inside function *shrink_page_list()*.

## Long-Term Goal

## Conclusion

- First
- Second