

Initial Block Download Disk and Virtual Memory Usage with libbitcoin v4

—

September 2019

Abstract

The report presents initial results for disk and virtual memory usage while running initial block download using libbitcoin v4. The results show that around after 300,000 blocks have been validated there is a slowdown in the rate of block validation and a clear sharp increase in resource usage.

1 Setup

We ran libbitcoin initial block download for about 6 hour period using AWS EC2 spot instance to save costs. We used the r5d.4xlarge instance that has the following specifications:

16 vCPU 16 cores on a virtual machine

128 GB RAM

2 x 300 GB NVMe SSD instance storage. This is not the normal EBS storage, but on host disk available to the instance. The advertised write speeds for the NVMe SSDs for r5d.4xlarge instances is 400Mbps [1].

10 Gbps maximum download speed available to the instance.

Ubuntu 18.04 with Linux kernel version 4.15.

1.1 Swap Space

Using the same instance type as above we also ran our experiment with a second setup that included 256GB of swap space. We used one of the instance stores available with the r5d.4xlarge instance and mounted it as swap for Linux. The instance store used is an on-host instance, i.e. it is not a network disk available as an AWS EBS.

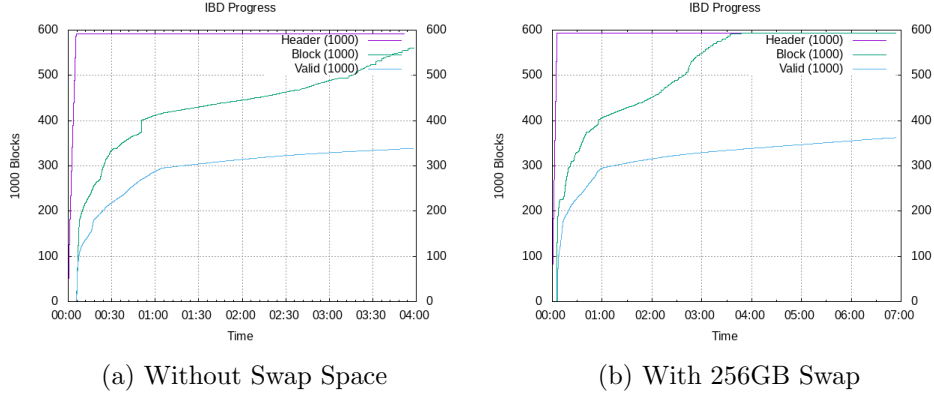


Figure 1: Initial Block Download Progress

2 Results

2.1 Progress

The figure 1a and 1b shows the block download and validation progress. The figure shows the headers are all downloaded in about first five minutes. Once the headers are downloaded the block download and validation starts simultaneously.

There is a slowdown in the rate of block download and block validation at around the one hour mark, or when around 400,000 blocks have been downloaded and close to 300,000 blocks have been validated. These numbers are valid for both setups of with and without swap.

Block validation in fact suffers two slowdowns in the rate of validation. When no swap is used, the two slowdowns are at around 100,000 blocks have been validated and then around 300,000 blocks. When an 256GB of swap space is provided, the slowdowns occur at around 200,000 and 300,000 blocks.

I think there is a well known reason for more validation work being done after 300,000 blocks. I can't remember what it is was. But it is clear something happens at around block 300,000. The results for the disk usage, memory usage and page faults shows how the resource usage increases around the 300,000 blocks. These results are shown next.

2.2 Disk Usage

The figures 2a and 2b show the amount of data being written to and read from the EC2 instance store by the node process. All the I/O for various threads is presented as an aggregate.

There are three major observations here. First, the disk write speed is far lower than the 400Mbps advertised by EC2. This is most likely because of

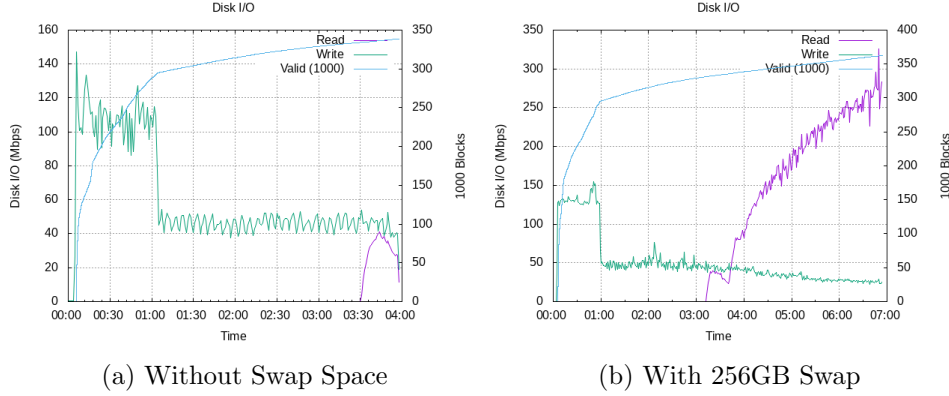


Figure 2: Disk Usage During Initial Block Download

the Linux filesystem buffers interacting with the VM paging system. Linux shares the filesystem buffers with the VM buffer manager, and an update of a memory location of a memory mapped file doesn't immediately get written to disk.

The other clear observation is the rapid drop in the amount of data being sent to disk by the process around the 300,000 block validation mark.

Finally, there is a sharp increase in data read from disk around the 300-325k block validation. In the first setup without the swap space, the process was stopped as we had hit the spot instance usage limit. However, in the case of swap (where the experiment ran for much longer) we see a really sharp ever increasing in data read from the disk. This could be because we need to access many diverse sections of the memory mapped file to validate transactions and blocks. It will be interesting to see how various blocks in the bitcoin blocks access different transactions from different blocks over time.

2.3 Memory Usage

Virtual memory usage shows how the we linearly increase the use of memory, but when we hit that point where disk reads take start to increase, the memory used by the process drops and then flat lines. This could be explained by increase demand by the OS for managing virtual memory.

2.4 Page Faults

Finally the page faults that requires memory buffers to be mapped in to the process address space (minor page faults) and sometimes even fetch data from disk (major page faults) are shown.

The increase in the major page faults directly maps to the period when data is read from the disk. Even if the number of page faults reaches a

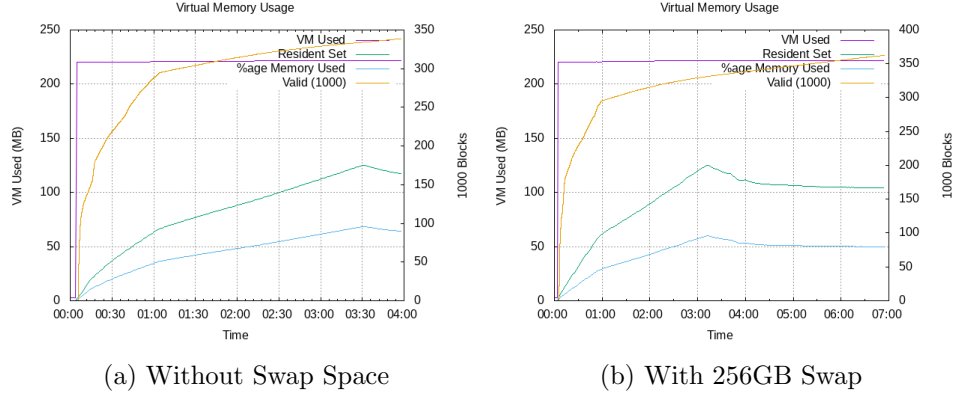


Figure 3: Memory Usage During Initial Block Download

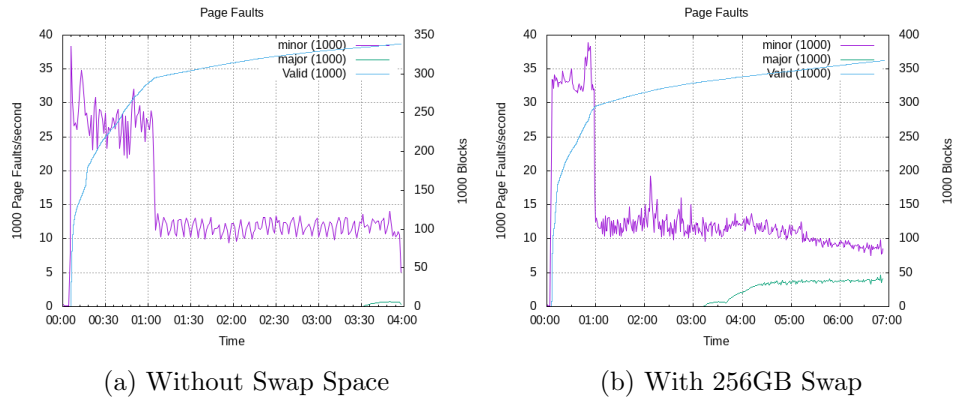


Figure 4: Page Faults During Initial Block Download

plateau, data read from disk per second keeps increasing. This could be a sign that there is some thrashing taking place as data is continuously read from and written to disk.

3 Conclusion

We presented our first effort to measure the system resources usage when running initial block download using libbitcoin v4. With systems and scripts in place we can run these experiments again to see how future work on libbitcoin improves the performance shown.

References

- [1] Ssd i/o performance. <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/memory-optimized-instances.html#r5d-z1d-instances-ssd-perf>.