CloudSSI: Revisiting SSI in cloud era

Mansoor Alicherry, Ashok Anand, Shoban Preeth Chandrabose, Theophilius Benson

1. MOTIVATION:

The current IaaS model has several shortcomings. First, several IaaS providers only offers VM (virtual machine) with predefined sizes, thus enterprise tenants must judiciously determine the VM size that best fit their application. This is challenging as overprovisioning VMs can lead to waste of resources while underprovisioned VMs can lead to poor performance. Second, when an application requires more resources than a VM can provide, tenants are currently limited to either scaling-out or scaling-up their applications. However, in both situations the granularity is at the level of VMs which leads to sizing issues discussed earlier. Third, scaling-up is ineffective as it incurs a significant amount of downtime/poor performance while the new VM is being provisioned and not all applications support scaling-out. For example while, Web servers can be easily scaled-out other legacy applications can not [1], thus limiting its applicability.

In this poster, we look at the problem of taking cloud into next level of flexibility, where applications can get resources as and when needed, and there is minimal wastage of unused resources. We make a case for leveraging the old idea of single system image (SSI) in the cloud context. With SSI, a process from an application can get resources (CPU, memory, and disk) from any of the VMs, and need not be constrained by the capacity of one VM. The legacy applications can run unmodified, and still use resources from multiple VMs. The processes can be seamlessly migrated to other VMs to avoid the network becoming bottleneck. Such flexibility would also allow packing processes efficiently into fewer VMs, and enabling enterprises to pay exactly for the amount of the resources required.

With the recent advances in reduction of network bandwidth and latency, we believe that SSI can help in providing such flexibility for cloud-based applications and applications need not be rearchitected. One of the limitations of SSI was scalability, however, we believe that many legacy apps don't need scalability to thousands of nodes, thus SSI can benefit such applications in cloud.

SSI can be realized in multiple ways: by changes at hypervisor, or OS, or middleware with different tradeoffs of implementation complexity, deployment ease and benefits.

2. CHALLENGES

To effectively realize SSI in the cloud, CloudSSI must overcome the following challenges:

Placement To effectively provide high memory bandwidth and low latency to VMs belonging to the same SSI, CloudSSI requires the cloud orchestrator to employ a VM placement strategy that places VMs close to each other (e.g., within the same rack). The main challenge in developing this placement strategy revolves around adjusting placement decisions to mirror the fact that the number of VMs in a SSI group is a function of the load.

Migration Migration should not affect performance of CloudSSI, so it may be desired to migrate VMs belonging to the same SSI together.

Failures VMs in cloud are prone to failures. These failures can propagate to multiple VMs in cloudSSI; for example, failure of a VM also affects the external process using remote memory from the VM. A potential way to deal with these issues is to keep the backups of remote memory pages in local disk; the hypervisor (or OS or middleware) should be made aware of these backups to retrieve from local disk in the case of failures.

3. PRELIMINARY EVALUATION:

We study the performance and cost benefits of applications using SSI-based approach. SSI enables us to obtain resources from any VMs. We find that even for 1 Gbps network link, using memory from a remote VM can increase the performance by 6X compared to local disk access. The performance is expected to improve further with links having higher bandwidth. We also evaluated for real applications: such as statistical package R, which loads full dataset in memory, and found performance benefits of 25%. We also find that by leveraging the ability of moving processes across VMs (using Mosix [2]), we could reduce the completion time by 32% for a transcoding application (FFmpeg). Finally, we find that SSI can help in using 30-60% fewer VMs for 25 application profiles (mix of memory-intensive and CPU-intensive applications), even with limiting use of remote memory to 25% of application demand.

REFERENCES

[1] The trouble with legacy apps. http://www.cloudswitch.com/page/the-trouble-with-legacy-apps, 2013.

[2] A. Barak, S. Guday, and R. G. Wheeler. *The MOSIX Distributed Operating System - Load Balancing for UNIX*, volume 672 of *Lecture Notes in Computer Science*. Springer, 1993.