Paper Review Form (1000 words maximum)
cs940: Container-based operating system virtualization: a scalable, high-performance alternative to hypervisors [1]

January 29, 2018

Paper Summary

3–5 sentences. Briefly summarise the **contributions** *of the paper, i.e., what it adds over the state of the art. Paraphrase and extract the essentials rather than simply copying chunks of text. Be objective; later sections allow for your own opinion.*

In this paper, the authors presented a thorough description of the container-based virtualization system Linux-VServer, which seeks to improve guest performance at the cost of reduced guest isolation and flexibility. Designed for use by compute farms and hosting organisations, VServer's container-based operating system (COS) isolates VMs at the system call layer, exposing the application binary interface (ABI) rather than a traditional hypervisor interface to guest systems. Resource and security isolation, as well as file system unification were implemented to achieve similar effects to that of a hypervisor. Benchmarks performed by the author found that VServer outperformed Xen in performance and achieved reasonable isolation.

Pros and Cons

6 bullets. Succinctly state three positives and three negatives of the paper.

I believe that the paper has the following positive features:

- Along with performance advantages identified in VServer's design, the authors did present some shortcomings in comparison with Xen for completeness, such as increased interface complexity of COS.
- Discussing the codebase, the authors evaluated both the number of lines of code *and* the context of these changes (new and modified kernel files), presenting a more complete picture of the complexity of the software-engineering task.
- Copy-on-write inodes was a novel idea taking full advantage of the straightforward file system offered by a COS, which is somewhat harder to achieve on file level on a hypervisor (e.g. Xen on LVM).

I believe that the paper has the following negative features:

- The argument that host VM in Xen-style hypervisors being the weakest link in fault isolation is somewhat questionable, as the large number of new and modified kernel modules in VServer are equally likely to contain bugs, in addition to a more grave concern of vulnerability in both systems.
- The micro-benchmarks performed by the authors were selectively presented in the table to highlight configurations where Xen performed significantly worse, with no additional information on overall scores.
- A majority of benchmarks performed did not account for a typical hypervisor/COS computation workload that would involve different guests utilising all of CPU, memory, and I/O. The only typical-like workload performed was kernel compilation, in which Xen achieved almost identical performance, drawing debate on the fairness of benchmarks.

The Problem/Motivation

1–2 sentences per question. What is the motivation for the work, or the problem being solved? Why is it important? If there is prior art, how was it insufficient? If the problem had not previously been solved, why not?

As a different approach to virtualization (and the definition thereof), container-based systems like VServer took a different direction in the performance-isolation tradeoff in comparison to hypervisors like Xen, whose paravirtualization variant while optimised for performance, does incur a non-negligible amount of overhead in CPU, memory, and I/O performance [2, Sec. 4]. Container-based systems seek to alleviate this through a different design, as summarised in the next section.

In addition, with a comprehensive set of benchmark results presented for Xen [2, Sec. 4], it would be useful to compare them with similar results from VServer to demonstrate the differences between the two paradigms, which will be carried out in this research.

The Solution/Approach

5–10 sentences. What have they done? How does it address the issues set out above? How is it unique and/or innovative (if, indeed, it is)? Give details, again using the paper as the source but again, not just copying text. Instead, focus on paraphrasing/synopsising, and extracting the essential details.

Based on prior research on resource containers and security containers, the VServer COS was designed with the goal of trading isolation for efficiency. As described earlier, the approach of container-based virtualization means that both the host OS and guest systems will operate on the same kernel, exchanging the flexibility in hosting foreign-kernel guests with simplicity of interface design and system call performance. Instead of Xen PV's approach of elevating guest kernels to ring 1 and filter unsafe system operations, VServer utilises a modified kernel to isolate operating system objects created by guest systems. Token bucket-based algorithms were implemented to ensure controllability and fairness in CPU and I/O scheduling, along with abilities to limit guest memory and storage use. Filtering and emulation were used to create a per-container illusion of dedicated system to its processes. Network interfaces were not virtualised, and Chroot Barrier was used to prevent chroot escapes. Finally, a copy-on-write mechanism was used on inodes to reduce storage footprint among commonly included guest OS files.

Evaluation

3–4 sentences. How do they evaluate their work? What questions does their evaluation set out to answer? What does their evaluation say about the strengths and weaknesses of their system? What is your opinion of the strengths and weaknesses of the evaluation itself? Give highlights, not a point by point reproduction of the evaluation section(s). In rare cases, systems papers may not have any evaluation, in which case write 'N/A' below.

The authors attempted to present comprehensive benchmarks similar to that by the Xen paper [2, Sec. 4], but the results were inadvertently heavy on micro-benchmarks. Both virtualization overhead and performance under disruptive guests were evaluated. Across micro-benchmarks and macro-benchmarks, a majority of VServer's imposed overhead in CPU, memory, and I/O were found to be lesser than Xen's. Owing to limited support for hyper-threading at the time, Xen could not achieve better scaling performance than VServer under highly parallelised workload as expected.

Results from isolation tests showed two systems roughly on-par, although the evaluation for accurate CPU reservation seemed to favour VServer by design, as in almost all use cases there are no practical disadvantage in Xen being 5% more off.

Your Opinion

At least 3 sentences. This is the fun part where you get to judge both the paper and the work it reports! Is the motivation convincing? The problem important? The approach a good or bad idea? Why? Which specific things annoyed you, or you thought were cool, or cool-but-flawed? Justify your opinions! Make an argument which will convince others of your opinion.

As a customer of virtual machine hosting for many years, container-based virtualization's ability for providers to vastly overcommitting resources caused many grievances in OpenVZ and VServer products. Despite this, I strived to be as neutral as possible in my review of this paper, which provided a very good point of reference in COS designs and VServer in particular. VServer's design has great potential in specific use cases when isolation is not of the greatest concern, such as rapidly deploying back-end servers in response to demand. In the virtual machine hosting market, VServer never really enjoyed much success compared to OpenVZ [3], due to several drawbacks, including the still unvirtualised network interfaces [4].

Questions for the Authors

Finally, imagine you're attending a talk about this paper given by one of the authors. Give at least 2 questions that you would like to ask, specific to the paper and the research it reports.

- Since the publication of this paper, what effects have the developments of multi-core and hyper-threaded processors had on your original design and evaluations of VServer?
- Given a history of security vulnerabilities in both hypervisors and COS [5], which paradigm of virtualization do you think can be more easily verified or otherwise audited for security?

References

[1] S. Soltesz, H. Pötzl, M. E. Fiuczynski, A. Bavier, and L. Peterson, "Container-based operating system virtualization: a scalable, high-performance alternative to hypervisors," in *ACM SIGOPS Operating Systems Review*, vol. 41, no. 3. ACM, 2007, pp. 275–287.

[2] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," in *ACM SIGOPS operating systems review*, vol. 37, no. 5. ACM, 2003, pp. 164–177.

[3] Y. Zhou, B. Subramaniam, K. Keahey, and J. Lange, "Comparison of virtualization and containerization techniques for high performance computing," in *Proceedings of the 2015 ACM/IEEE Conference on Supercomputing*, 2015.

[4] L. VServer, "Networking vserver guests," October 2011. [Online]. Available: http://linux-vserver.org/Networking_vserver_guests

[5] V. Costan, I. Lebedev, S. Devadas *et al.*, "Secure processors part ii: Intel sgx security analysis and mit sanctum architecture," *Foundations and Trends® in Electronic Design Automation*, vol. 11, no. 3, pp. 249–361, 2017.