Secure Enclaves: An Isolation-centric Approach for Creating Secure High-Performance Computing Environments*

[Extended Abstract]

Ferrol Aderholdt, Susan Hicks, Thomas Naughton† and Lawrence Sorrillo
Oak Ridge National Laboratory

Blake Caldwell
University of Colorado Boulder

James Pogge and Stephen L. Scott
Tennessee Technological University

Categories and Subject Descriptors

H.3.4 [**INFORMATION STORAGE AND RETRIEVAL**]: Systems and Software—*Performance evaluation (efficiency and effectiveness), Distributed systems*; K.6.5 [**MANAGEMENT OF COMPUTING AND INFORMATION SYSTEMS**]: Security and Protection

General Terms

Information Systems, Computing Milieux

Keywords

HPC, Cloud, Virtualization, OpenStack, Lustre

1. OVERVIEW

High-performance computing (HPC) environments are used for a wide variety of workloads including simulation, data transformation and analysis, and complex workflows. When processing data at various security levels, the system is often enclaved at the highest security posture, which may limit usability or performance. The traditional approach used to provide isolation is effective at the creation of secure enclaves, but poses significant challenges with respect to the use of the shared infrastructure in HPC environments.

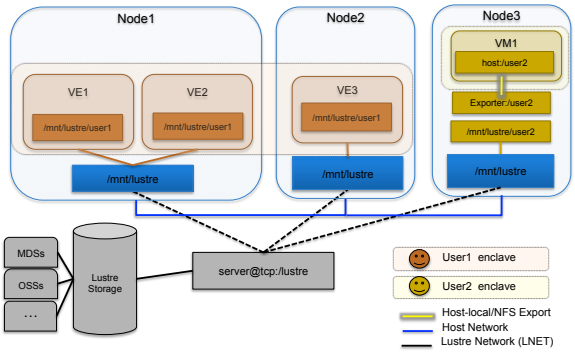


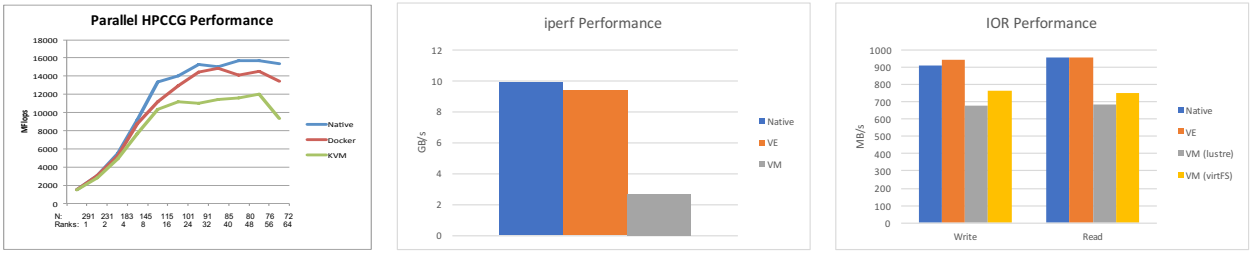
Figure 1: Isolation-centric architecture

In this poster, we evaluate the use of system-level (i.e., hypervisor-based) and operating system-level (i.e., container-based) virtualization as well as software defined networking (SDN) as possible mechanisms for secure, isolation-centric enclaves (secure enclaves). In these secure enclaves, isolation is provided with respect to (i) compute, (ii) network, and (iii) data storage. The compute layer is isolated through the use of system-level and operating system-level virtualization (i.e., virtual machines (VM) and virtual environments (VE), respectively), which provides users with varying levels of customizability while maintaining high-performance. The network layer is partitioned at runtime through the use of SDN, which enables isolated communication with multiple tenants while maintaining high-performance between the nodes within the system. The isolation of high-performance data storage is provided through the abstraction of the mount location given to the VM or VE at boot time. An example of this architecture can be seen in Figure 1.

To evaluate this architecture, we make use of microbenchmarks evaluating each layer's performance, while providing isolation between users. While the effectiveness of these mechansisms with respect to providing isolation and secure compute environments has been shown in previous work, this work evaluates the effectiveness and performance of these approaches with HPC-centric microbenchmarks. We have setup a prototype of this work, which can be seen in Figure 1. We demonstrate the effectiveness of this work through the use of microbenchmarks meant to evaluate the overhead provided by the isolation mechanisms with respect to the performant capabilities of the system.

†Contact author: Thomas Naughton (naughtont@ornl.gov)

Supercomputing 2017 Denver, Colorado USA



(a) (b) (c)

Figure 2: The experimental evaluation of (a) compute, (b) network, and (c) data storage

2. BACKGROUND

The isolation mechanisms are the basic operations required for enabling secure enclaves at multiple levels including compute, network, and data storage. Here we will breifly discuss the background on these mechanisms.

With respect to compute isolation, virtualization is typically used and may be categorized as either (i) system-level or (ii) operating system-level. In (i), the entire execution environment is abstracted and isolated within a VM that is executing on a hypervisor (e.g., Xen, KVM, etc.), which gaurantees the underlying ABI of the system. In (ii), the exectuion environment with respect to applications and required libraries are isolated within a VE that is executing on a host system with the host OS gauranteeing compute and memory isolation (e.g., *cgroups* and *namespaces* in Linux). We explore both approaches for our secure enclaves, which allows for flexibility with respect to user environments.

To provide network isolation, SDN is used. SDN allows for the creation of reconfigurable networks while maintaining scalability through well-defined APIs. This can allow for multiple enclaves to be deployed while their networked communication will be isolated. The ability to provide isolation of communication and reconfigure this isolation during runtime has caused many management systems such as OpenStack to incorporate the functionality necessary to communicate with SDN enabled network switches and routers.

Data storage in HPC environments is often made available via a parallel filesystem (e.g., Lustre, GPFS, etc.) to provide high-performance data reads and writes. With respect to data storage, there are two key areas in which isolation is required: (i) data in flight and (ii) data at rest. In (i), the data being pushed to the multiple disks will have to be sent over the network, which should be isolated through SDN. For (ii), data access would need to be restricted to sub-trees that are currently at the user's permission level. With respect to the Lustre filesystem, both of these areas of isolation are possible. For (i), aside from the usage of SDN, Lustre Network (LNET) configurations may be used to restrict access to particular NICs, which can be used to hide the filesystem from a guest such that access is provided through the host. For (ii), the ability to mount sub-directories by a client has been available in Lustre for over a year. Additionally, virtFS [1] may be used to provide IO-forwarding access for a VM to the data storage while restricting the overall access to a particular sub-tree.

3. EVALUATIONS

To evaluate the system, we used multiple microbenchmarks and focused on the three areas of compute, network, and data storage to understand the overhead of the isolation mechanisms and their impact on each stage. For the evaluation, the microbenchmarks used include HPCCG (compute), iperf (network), and IOR (network and data storage) to a Lustre filesystem. The testbed for the evaluation was an internal test cluster with 10 Gb Ethernet interconnect. For operating system-level virtualization, we made use of Docker using a host network; for system-level virtualization, we use KVM using the Intel e1000 driver. The iperf results show raw performance without a SDN, while the SDN was included in the IOR results. This is to demonstrate the overhead on the network with and without isolation. The results can be found in Figure 2.

4. CONCLUSION

Operating system-level virtualization shows great promise for providing efficient and secure compute enclaving. The evaluated benchmarks have shown near native performance with respect to compute, network, and data storage. System-level virtualization through KVM can achieve good performance, but has significant overhead, particularly with respect to network performance. We are able to leverage OpenStack and SDN to achieve on-demand network enclaving, as demonstrated on SE testbed deployed at ORNL. SDN provides the functionality necessary to configure distributed networking components on-demand, while at the same time providing desired performance, security, and reliability goals. Applying these virtualization technologies for HPC provides an increased level of flexibility with security and performance. The ability to have dynamically reconfigurable networks enables per-tenant allocations, which greatly increases the ability to isolate applications (tenants) within a shared infrastructure.

5. ACKNOWLEDGEMENTS

We would like to thank Galen Shipman for his leadership and contributions to the Secure Enclaves project while he was at ORNL.

This work was supported by the United States Department of Defense (DoD) and used resources of the Computational Research and Development Program at Oak Ridge National Laboratory.

6. REFERENCES

[1] V. Jujjuri, E. Van Hensbergen, A. Liguori, and B. Pulavarty. VirtFS—A virtualization aware File System pass-through. In *Ottawa Linux Symposium*, pages 1–14, Dec. 2010.