Cluster Security Research Challenges

William Yurcik[†] Adam J. Lee[†‡] Gregory A. Koenig[†‡]
Nadir Kiyanclar[†‡] Dmitry Mogilevsky[†‡] Michael Treaster[†‡]
[†]National Center for Supercomputing Applications (NCSA)
[‡]Department of Computer Science
University of Illinois at Urbana-Champaign
{byurcik,adamlee,koenig,nadir,dmogilev,treaster}@ncsa.uiuc.edu

ABSTRACT
In this paper, we share insights from our group experience building and experimenting on high performance computing clusters to support our research developing novel cluster security protection techniques and tools.

Categories and Subject Descriptors
C.1.4 [**Processor Architectures**]: Parallel Architectures – *distributed architectures*. C.2.0 [**Computer-Communications Networks**]: General – *security*. C.5.1 [**Computer System Implementation**]: Large and Medium (``Mainframe") Computers – *super (very large) computers*. K.6.2 [**Management of Computing and Information Systems**]: Installation Management– *computing equipment management*.

General Terms
Experimentation, Management, Measurement, Security

Keywords
cluster security, high performance cluster computing, cluster management, research challenges, lessons learned

1. INTRODUCTION
Our research group has been relatively successful at combining cluster computing research with security and traditional computer science (parallel programming and computer architectures) [2,4,5]. However, this success does not mean there are no challenges, in fact just the opposite. There are many challenges and we hope by sharing our lessons others may learn from our experience - we certainly owe a debt of gratitude to lessons we have learned from other researchers. We generalize our experiences into two groups: (1) building/managing a cluster and (2) performing experiments. We note for completeness there are different types of HPC clusters as described in [2] so our insights need to be filtered for applicability.

NSF/CISE/CNS Cluster Computing Infrastructure Experience Workshop, July 27, 2005, Siebel Center for Computer Science, University of Illinois at Urbana-Champaign, IL., USA.

2. INFRASTRUCTURE CHALLENGES
Building a high performance computing (HPC) cluster system requires a significant investment in hardware/software selection and implementation [1]. Often, the equipment is "hand-me-down" equipment - this has been our scenario. While the assembled hardware confined us in terms of the configuration, it also provided an initial trusted setup to build upon. However, there were also peculiarities in equipment failures and time spent finding drivers to support outdated equipment.

When deciding which software to install, our choices were driven by a desire to balance configurability with ease of administration. Our projects require hosts running standard Linux facilities, along with a standard suite of tools for distributed computing such as MPI. We also require the ability to install research software and make kernel modifications.

We chose the ROCKS cluster toolkit because of its ease of installation [3]. This choice placed some constraints on system configuration since ROCKS enforces the notion of an 'enclosed cluster' consisting of a head node and multiple compute nodes. For on-demand computing experiments in which we execute distributed jobs across clusters, this must be circumvented. Overall, this decision has worked well allowing us to stop malicious network traffic at a single chokepoint and minimizing IP address administration.

Setting up research oriented clusters can be challenging – it is an advanced task. We were particularly challenged by setting up PXE boot sequences correctly on the nodes as well as being unable to get the nodes to boot the installation from the head node automatically. Maintenance will require human resources since it is an on-going process that does not end when the cluster software installation is finished, a cluster requires require many adjustments to remain usable.

Lastly, researchers may use weak passwords or shared accounts. This requires someone to serve as a dedicated system administrator to monitor system log files and "root" e-mail since a compromise on a research cluster can negatively impact the wider organizational community.

3. EXPERIMENTATION CHALLENGES
A research-oriented cluster differs from typical production clusters. In research clusters, many users have "root" privileges since it is easier to modify the environment yourself rather than going through a system administrator. The downside is the research cluster environment generally tends toward chaos with

lack of accountability. For example, there may be several different (possibly even competing) versions of the same software installed.

The benefits of building and maintaining a research cluster for experimentation are worth it. You do gain exclusive access for testing software under development. This is especially useful for system level software versus user-level scientific software because system-level software faults can disrupt an entire cluster. There are also benefits in scheduling grid or on-demand jobs across multiple clusters simultaneously. In a production environment, you usually have to schedule 2-4 hour blocks of time a week in advance not to mention that you may not be able to debug your problems in such a small window of opportunity. In contrast, research clusters tend to be a "free for all" regarding resources, with much less emphasis on things like job schedulers to allocate nodes to individual jobs.

Our group is exploring the development of patches to the Linux kernel to aid in cluster security monitoring. From the perspective of production clusters, it is clearly a problem to allow parties, even privileged ones, to load kernel modules or otherwise modify the kernels of cluster machines. One potential solution was to go with a completely user-space solution to both problems, the primary candidate in this direction being User-Mode Linux (UML).

UML documentation, as well as our own experiments, convinced us that this solution is not suitable for performance and security reasons. UML showed poor performance, relative to Linux, on native hardware in our experiments (a result which has been widely confirmed). This is because *every* system call from a hosted user mode instance of the kernel must be trapped using the Linux *ptrace* interface. Furthermore, the UML kernel must emulate many system calls, including those involving IO to virtual block devices backed by real hardware, by making system calls into the real kernel, conservatively incurring a 2-to-1 performance penalty. UML is also insecure when running on an unmodified hosting kernel due to the fact that system is implemented by placing a copy of the 'virtual kernel' in the upper address space of all of its 'virtual' user-space processes. Malicious processes can thus read and modify the memory space of the UML kernel. These security issues may be addressed by patching the hosting kernel to support a new Separate Kernel Address Space (skas) mode. However, kernel patching would have defeated the purpose of a completely user-space goal.

During experiments on developing a fault-detection API, several things became apparent. The current MPI implementation is low level, perhaps too low level. It is difficult to predict the behavior of a multi-threaded program in which multiple threads attempt simultaneous MPI calls. MPI is also ill-suited for environments with no QoS guarantees. For example, MPICH2 has undefined behavior when MPI_Finalize is called when there is an unresolved MPI_Ireceive call still present. Another example is calling MPI_Bcast, yet failing to receive the broadcast due to network or node failure.

4. SUMMARY

In summary, while studying security in cluster computing we have learned two primary lessons:

Lesson 1: Studying security in HPC clusters is *harder* than studying security in enterprise networks. To effectively study the security of clusters, researchers must have full access to explore the environment in a realistic setting. The emergent properties of clusters make studying security difficult to simulate, as working with a small number of nodes does not allow researchers to fully explore the relationships present throughout a system as a whole. Full access to a production clusters is difficult to attain, as members of the HPC community are reluctant to experiment without guarantees and immediate benefits. For these reasons, cluster security researchers must experiment on their own cluster(s) – a significant barrier.

Lesson 2: Studying security in HPC clusters is *easier* than studying security in enterprise networking. A large percentage of the overall cluster nodes may be partitioned into a small number of equivalence classes. This allows researchers to make simplifying assumptions (that cannot be made in enterprise network environments) regarding homogeneous communication patterns, configurations, running services, and other aspects of the environment. In addition, the networking that connects cluster nodes is typically physically and/or logically centralized. Lastly, a cluster environment is more constrained than typical enterprise environments in terms of software, processes, and users making security experimentation an easier task.

These seemingly contradictory lessons lead to an important conclusion: studying the security of HPC clusters is fundamentally different than studying the security of enterprise networks. **In order to make substantial progress on this front, it is important that the HPC and security communities embrace one another's work and form collaborative relationships so that both communities can prosper.** Without the support of the HPC community, security researchers find it difficult to fully explore their ideas. Likewise, without the support of the security community, the HPC community will likely continue to be plagued by security incidents on the scale of the TeraGrid compromises during the Spring of 2004 which has taken over a year to fully investigate.

5. REFERENCES

[1] Paz, M.B., and Gulias, V.M., "Cluster Setup and its Administration," Chapter Two within the book *High Performance Cluster Computing Volume 1: Architectures and Systems* (edited by R. Buyya), Prentice Hall, 1999.

[2] Pourzandi, M., Gordon, D., Yurcik, W., and Koenig, G.A., "Clusters and Security: Toward Distributed Security for Distributed Systems," *IEEE Cluster Computing and Grid (CCGrid),* 2005.

[3] ROCKS Cluster Distribution, <http://www.rocksclusters.org/Rocks/>

[4] Yurcik, W., Koenig, G.A., Meng, X., and Greenseid, J., "Cluster Security as a Unique Problem with Emergent Properties: Issues and Techniques," *5th LCI International Conference on Linux Clusters,* 2004.

[5] Yurcik, W., Meng, X., and Kiyanclar, N., "NVisionCC: A Visualization Framework for High Performance Cluster Security," *ACM CCS Workship on Visualization and Data Mining for Computer Security* (VIZSEC/DMSEC), 2004.