TECHNICAL DOCUMENT 3430
JANUARY 2024

Building the Evolving and Flexible High Performance Computing Enterprise

Stephen T. Pease
George Z. Istambouli
Ziad A. Haddad
Daniel J. Arevalo
Daniel W. Phillippi
NIWC Pacific

Naval Information Warfare Center (NIWC) Pacific
San Diego, CA 92152-5001

This page is intentionally blank.

TECHNICAL DOCUMENT 3430
JANUARY 2024

Building the Evolving and Flexible High Performance Computing Enterprise

Stephen T. Pease
George Z. Istambouli
Ziad A. Haddad
Daniel J. Arevalo
Daniel W. Phillippi
NIWC Pacific

Administrative Notes:
This document was approved through the Release of Scientific and Technical Information (RSTI) process in January 2024 and formally published in the Defense Technical Information Center (DTIC) in January 2024.

NIWC Pacific
San Diego, CA 92152-5001

NIWC Pacific
San Diego, California 92152-5001

P. M. McKenna, CAPT, USN
Commanding Officer

M. J. McMillan
Executive Director

ADMINISTRATIVE INFORMATION

The work described in this report was performed by the Command & Control (C2) Systems Division (Code 53236) of the Command & Control and Enterprise Engineering (C2E2) Department, Naval Information Warfare Center (NIWC) Pacific, San Diego, CA.

Released by
Christopher Johnson, Division Head
Command and Control Systems Division

Under authority of
Scott Crellin, Department Head
Command and Control Enterprise Engineering Department

ACKNOWLEDGMENTS

Editor: MGK

ACRONYMS

AI	Artificial Intelligence
API	Application Passing Interface
BMC	Baseboard Management Controller
CAC	Common Access Card
CPU	Central Processing Unit
CXL	Compute Express Link
DAS	Direct Attached Storage
DISA	Defense Information Systems Agency
FPGA	Field-Programmable Gate Array
GPU	Graphics Processing Unit
HPC	High-Performance Computing/Compute
HPCC	High Performance Computing Center
I/O	Input/Output
LLM	Large Language Models
ML	Machine Learning
MSA	Modular Storage Array
NFS	Network File Storage
NIWC	Naval Information Warfare Center
NVMe	Non-Volatile Memory Express
OS	Operating System
PCIe	Peripheral Component Interconnect Express
RHEL	Red Hat Enterprise License
SAN	Storage Area Network
SoS	System of Systems
SSD	Solid State Drive
SSH	Secure Shell
STIG	Security Technical Implementation Guide
VM	Virtual Machine

This page is intentionally blank.

CONTENTS

ACRONYMS ..... **V**
1. INTRODUCTION **1**
1.1 PURPOSE 1
1.1.1 Objectives 1
1.2 BACKGROUND 2
2. MANAGING RESOURCES AND TEAMS **3**
2.1 RESOURCE MANAGEMENT 3
2.1.1 The Online Repository 3
2.1.2 Team Management 4
3. TAILORING APPLICATIONS **7**
3.1 CHANGING THE FACE OF THE SYSTEM 7
3.1.1 Open OnDemand 8
3.1.2 Operating System (OS) Standardization 10
3.1.3 Ansible 10
3.1.4 Nagios Core 10
4. SYSTEM HARDWARE AND VIRTUALIZATION **11**
4.1 STORAGE 11
4.1.1 Storage Area Network (SAN) 11
4.1.2 Network File Storage (NFS) 11
4.1.3 Parallel File System 11
4.2 NETWORK 12
4.3 VIRTUALIZATION AND INTERCONNECTIVITY 13
4.3.1 Server Virtualization 13
4.3.2 Device Interconnection 14
5. CONCLUSIONS **17**

FIGURES

Figure 1. The HPCC online repository. 4
Figure 2. The old MobaXterm interface for SSH access. 7
Figure 3. The new web interface for system access. 8

This page is intentionally blank.

1. INTRODUCTION

1.1 PURPOSE

High-performance computing (HPC) system design, construction, and management is already a challenging task with numerous dependencies. However, that complexity is raised exponentially when factoring in the requirement to be flexible with ever-evolving technological changes and breakthroughs. A command must have systems that are modern and relevant to the work that their command is most interested in performing. This document explores the successes and suggestions for developing a system that maintains its place on the edge of technological development while providing flexibility to pivot into new areas or allow for changes to be made rapidly and efficiently, ensuring the mission continues to be met and exceeded.

In this exploration of our own development, we will focus on three key areas for building a meaningful system of systems (SoS) that houses the power and malleability to keep pace with current and future workloads as they present themselves across an enterprise. These key areas are:

- Managing Resources and Teams
- Tailoring software applications and authentication
- System Hardware and Virtualization

There will undoubtedly be an unending list of additional addressable items; however, this document seeks to scope the guidance to best practices that enable a flexible environment rather than a rigid step-by-step build of a system that is outdated as soon as it is finished.

1.1.1 Objectives

In designing these systems over time, we found that there were key objectives that needed constant focus, including flexibility, usability, capability, and consistency. These will be key ideas throughout each section as we discuss the improvements in design for our systems.

1.1.1.1 Objective 1 Flexibility

Flexibility was one of the most important items for our system development because requirements change, funding becomes available or is reduced, and technology improvement is a constant driving force. We found that just building powerful servers wasn't enough to bring in new developers, nor was it enough to garner interest in utilizing the systems. We designed our systems to enable multiple ways of accessing them and remain open to rapidly accepting approved software packages and applications.

1.1.1.2 Objective 2 Usability

Usability was another driving force behind improving usage and, more importantly, deeper collaboration between teams at the enterprise level. By providing systems that were simpler to use and navigate, developers that had critical skill sets in software but were more novice in nature toward using HPC systems were able to comfortably work in collaborative environments with more seasoned HPC users and begin to familiarize themselves with more powerful platforms.

1.1.1.3 Objective 3 Capability

Capability is one of the fundamental needs of an HPC system, because without heavy processing power and speed in data transfer to meet the needs of modern data sets and requirements, there is no

added benefit beyond using a personal computer for development. System capability was built from the inputs of users and collected monitoring data that told the story of what users were working on so that any limitations were addressed at the physical level through procurements to meet the needs of the current workforce with user interaction that told the needs of the future.

1.1.1.4 Objective Consistency

Another key improvement in our systems was consistency, because users do not want to leverage one capability only to find that when they pivot to another software or environment, the interface has completely changed and halted their development progress. The point is to move fast, and by setting the interface for access to our systems, users are met with a consistently familiar work environment that is easily navigable, requiring little to no instruction even for the most novice developers.

1.2 BACKGROUND

The High Performance Computing Center (HPCC) at the Naval Information Warfare Center (NIWC) Pacific had been mildly faltering for a while with minimal support, difficult usability, low user interaction, and a lack of consistent compliance. Our core team took over the HPCC at roughly the same time; however, nobody had a great deal of HPC experience, and we were a collection of broad skillsets consisting of network experts, mathematicians, application builders, and physics. What was initially viewed as a potential concern due to a lack of first-hand experience rapidly became a strength, as it made it easy for the team to adjust to new ideas and not fall into the “We’ve always done it this way” pitfall.

Beginning with bringing systems online, the team needed to rapidly determine where the baseline functionality was suffering, preventing even the most advanced users from engaging with the systems. The team's first modest success, finding methods to ensure the systems' reliability and compliance, marked the beginning of a journey from a lengthy, dark tunnel towards light.

The first few months and even years yielded a relatively painful learning curve, but the team came together to share lessons learned, come up with new ways of approaching problems, and search for any form of automation and methods of making things faster or easier. The sections forthcoming will explore the lessons learned, potential future developments, and what worked to create a welcoming environment open to all levels of expertise for the command.

2. MANAGING RESOURCES AND TEAMS

2.1 RESOURCE MANAGEMENT

Managing resources is certainly difficult; however, it tends toward the more innate capability of most technical personnel. Everyone has a method to their madness or a "system" that they use to maintain and manage versions, documents, time, schedules, etc. However, not all methods are the same, nor do they cleanly align with one another, so the first thing that we learned as a team was how to pool our expertise in broad areas and collectively come together for the management of our resources. The first thing we changed was hard copy versioning and tracking. There were papers littering the workspace detailing disjointed processes with nothing cleanly laid out, and searching was very literal in the sense that one would need to dig through piles of paperwork in order to find guidance or past lessons learned.

Beyond processes, we needed a home for rapid access to commands, tricks, techniques, efficient pathways for standing up systems, and generating a clean trail of breadcrumbs that got us to where we are today so at any point we could stand up everything from scratch. Same problem, but with different resources, and we needed a place to store our artifacts.

Further, we needed to have awareness of what was actually happening on our systems without needing to pester our users endlessly, wasting both their time and our own, when the tools exist to monitor usage and a simple question or two in chat could provide the missing puzzle pieces that would help determine our future procurements and system design.

Finally, in resource management, we needed to be cautious with our time as a team. The available hours in a workday are finite, and though we want everything to get completed yesterday, the reality is that our small team was capable of only so much within the hours we had, so they needed to be utilized efficiently for maximum effect.

The main concepts on which we focused for the improvement of our resources were:

- An online repository where all artifacts could be collected and updated collectively.
- Openly discussing time resources available across the team.

2.1.1 The Online Repository

Paperwork, even when we took over the HPC, should not have been an issue since the technology for online digital artifacts had been available for quite some time, but we were faced with that reality and developed a repository (see Figure 1) that would always be connected to our systems, enabling us to add and amplify anything that we had come across, completed, or needed to work together to complete.

This was created on our internal server so that everything was behind our internal firewall and detailed a broad range of items, including how to address challenging Security Technical Implementation Guide (STIG) remediations, code examples for rapid deployment of environments, scripting examples for user creation or automation of system hardening, and even key points of contact for various teams at the center that could support any questions we had during our build and maintenance phases.

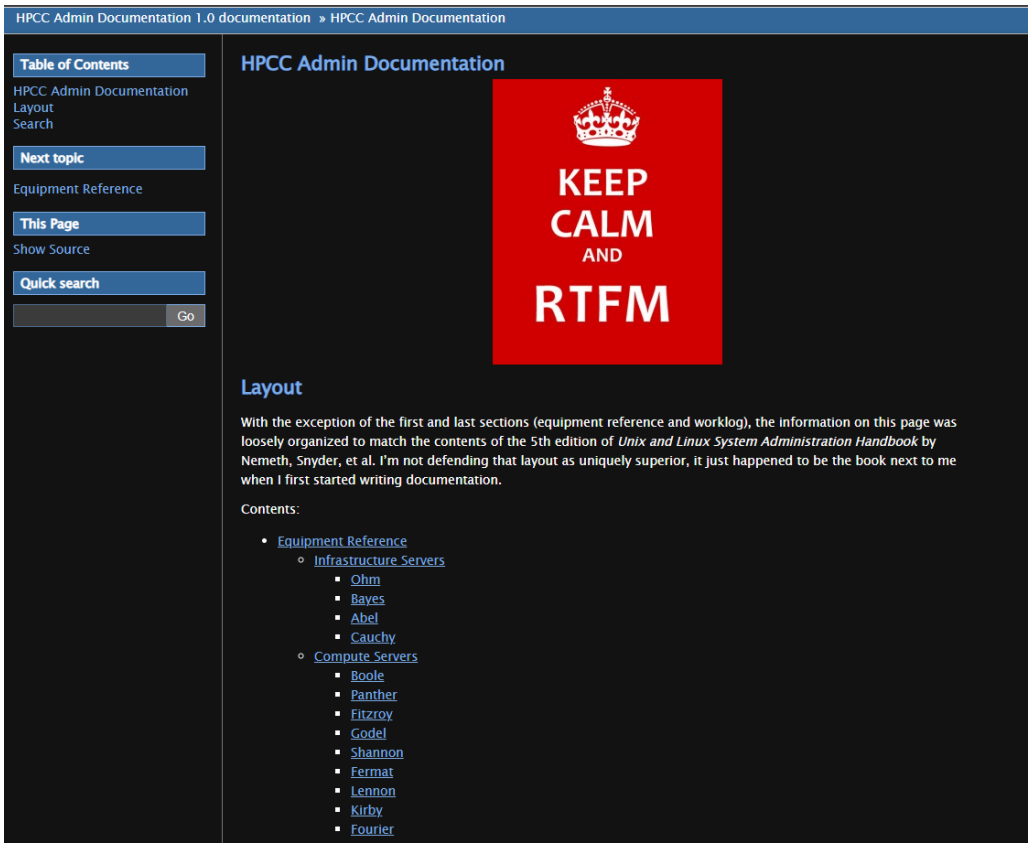

Figure 1. The HPCC online repository.

A pleasant consequence of standing up this repository was that after collecting all of these tips and quick reference code lines and scripts, we were able to selectively release access to the entire command, providing tailored rapid cybersecurity hardening techniques for groups across the center that could reuse our findings to more rapidly secure their systems, understand what was needed, and, most importantly, start doing work sooner, unencumbered by painfully slow research that was previously unshared and unavailable.

2.1.2 Team Management

A key term that has followed the group through a number of different roles and experiences and has benefited nearly every role is “strategic failure.” A team leader needs to have the comfort of breaking to fix and sometimes simply to better understand a system. We were in a unique position, in that we were all fairly novices coming into this world of HPC, but that afforded no resounding voice aside from the final say from management. This final voice was less of an authority and more of a thoughtful delegator between opinions and facts. We all shared what we knew and what we thought and presented all our ideas, regardless of how wild or outlandish they were. The only level-set that was required was the ability to maintain awareness of the command-level mission, so there is always a guiding star to maintain a general course, but how we got there was inventive, unique, and constantly changing along with technology, efforts, and realizations of how things could be better.

We set clear roles that team members held ultimate responsibility over; however, anyone who had an idea or question about a particular area was free to speak up, suggest a better way, and assist in all areas. By adopting this mentality, the collective group became stronger in all aspects of HPC over time, and systems grew from the opinions and creativity of all rather than one. Holding weekly

meetings, we would invite stakeholders to tell us what they need, talk about their projects, and offer solutions to problems they came across and resolved, so that we are always taking in as much information as possible while addressing our own internal development. We also ensure that we maintain a constant learning environment by leveraging professional training courses that are hands-on. These courses typically take us out of our comfort zone and focus on things outside our expertise, then procure enough seats to offer some to the command. This resulted in numerous partnerships, system improvement discussions, and new project ideas to support the mission in a more collaborative group setting. Key here is actual hands-on, leveraging digital online representative environments, because everyone has had enough PowerPoint.

All of these methods continue today, even as roles have become more defined. Team members always speak up about possible ideas and discuss how seemingly impossible things might be possible; failure is expected instead of avoided, but they always provide new information and better ways of approaching challenges and solutions. Nobody on the team should be wary of saying, “I don’t know” or “Let me find out.” Building from an engrained team that has a single voice on top into a functional and collaborative team may take time to dismantle the “wait for orders” mentality, but the benefit of working toward the “final voice of moderation” will instead create a team that is flexible, learns rapidly, works with others regularly, and is willing to provide ideas, resulting in continual improvement of systems and teams.

This page is intentionally blank.

3. TAILORING APPLICATIONS

3.1 CHANGING THE FACE OF THE SYSTEM

Part of the initial difficulty in leveraging the power of the HPC systems, particularly with novice-level users or even those with no experience, is that access to the systems was achieved through the command line. This means users required some basic understanding of the Linux command line; in addition, typically, most would need to install a program that enabled additional functions in order to access applications through our systems. In our case, that meant installing MobaXterm (see Figure 2) to allow secure shell (SSH) access. This provided its own unique challenge, as admin access was required for many user stations to install that program, or a wait time dependent on how fast a system administrator could access their system and push the software to them.

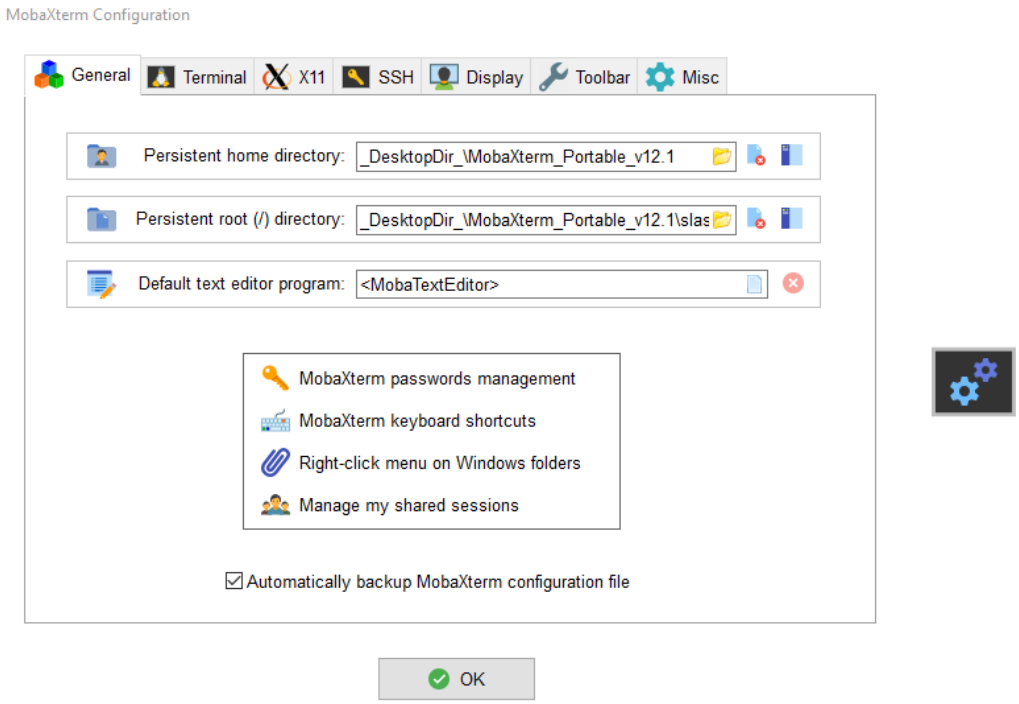

Figure 2. The old MobaXterm interface for SSH access.

In addition to the clunky method of accessing the systems, authentication was difficult and required that the team maintain records for access on each individual system, requiring users to follow a lengthy process to provide their Common Access Card (CAC) certifications in order to authenticate their access to the systems or rely on a regularly expiring username/password combination for each system.

A change was needed that addressed numerous issues:

- We needed to increase usability and invite a broader range of users.
- We needed a better, more automated authentication system.
- We needed to remove the learning curve of the required command-line knowledge.

3.1.1 Open OnDemand

While constantly working toward automating processes and making small strides in improvement by scripting authentication, combining login instances to allow multiple systems to leverage the CAC, and running programs in the background to reduce effort in access, it wasn't until we discovered Open OnDemand that we attained a massive leap forward in capability and usability. Conceived in 2007 at the Ohio Supercomputing Center by a collaborative group between the University of Buffalo, the University of Colorado Boulder, and Virginia Tech, Open OnDemand debuted in 2013 at the Extreme Science and Engineering Discovery Environment conference. Starting small, the web portal interface has gained massive traction and grown from 5 known installations in 2017 to currently over 250 installations at numerous HPC centers.

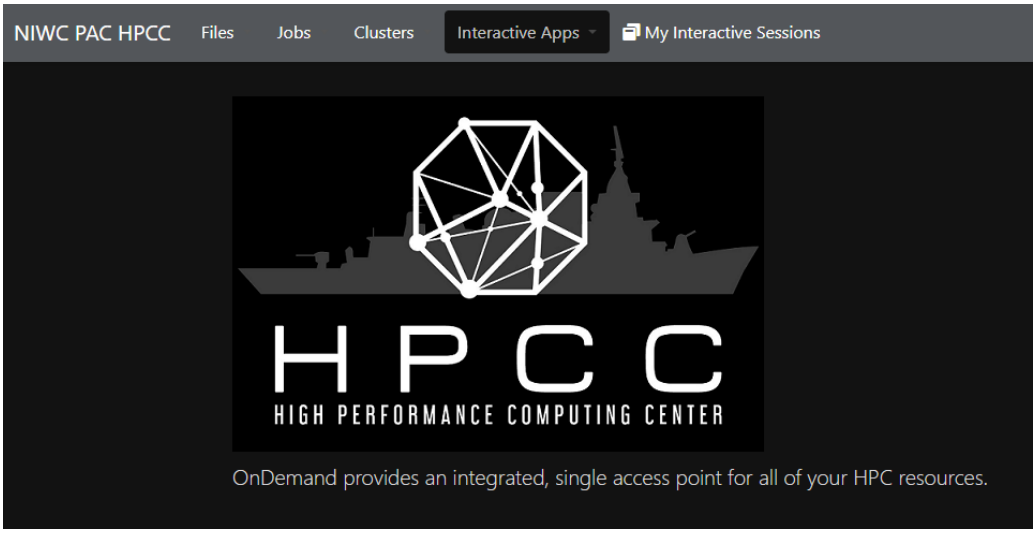

Figure 3. The new web interface for system access.

We initially brought this web interface functionality into our testing environment in 2021 and began running through multiple iterations to determine the best implementation for our users and potential new users that were unfamiliar with the HPC environment. Since then, our fully released version (see Figure 3) has exponentially increased our user base from a few users to over 400, while maintaining ease of access, better usability, application integration, and little to no administrative burden on users.

3.1.1.1 Usability

Results spoke for themselves as user count soared and continual use soared. We were no longer seeing groups log on to look; rather, we were seeing groups log on to use and reuse, again and again. We no longer only had seasoned command-line users using secure shells to access our systems; we had newer generations of developers making use of our tools and organically helping us to develop new applications that benefitted their use-cases, but we further built up our available tools. This malleable interface provided not only a place to do work but also a way to generate environments that were customized to increase the speed at which work could be done.

Particularly of interest to the artificial intelligence (AI) community, high-powered systems are a requirement; however, many simply use the systems and have them installed on their machines, preventing easy collaboration between teams. By producing template versions of their environments,

we saw a rapid influx of new technology development and engagement across teams. This ability to better share information produced rapid development and delivery of capability to projects that otherwise simply did not have the resources and, more to this point, did not have the capability to generate these environments at this scale.

This additionally brought in interested personnel that had no prior experience because it was so easy to use and explore. We provided training modules for simple ways to code, build models, and use particular applications, and accompanied those modules with step-by-step walkthroughs and videos. This was critical because it brought a new perspective to technology. Those with different technical backgrounds and experience enable broader thinking about how technology can be better developed and avoid failing into the "same as we have always done it" way of thinking about solving problems.

3.1.1.2 Authentication and User Management

Systems need to not only be easy to use, but they also need to be easy to access. That does not mean easy access for anyone, especially in the government and in high-profile corporations; however, approved and validated employees should not have trouble working on their own internal efforts. The administration burden also needs to be low, enabling a lean team to manage and maintain the infrastructure. In order to accomplish this, we began with a script to retrieve user information (ID, username, full name, email, code, etc.) from our command's centralized Lightweight Directory Access Protocol server, which ensured that we were in lock-step with the command's approval of who was allowed to access any systems.

The script then adds these users to all systems, ensuring a consistent user profile across all servers. Disparate group IDs affect data access for our users. We unified the group IDs across all servers, which allowed them access to group data within shared group directories. We then wrote an Ansible playbook (described in 3.1.3) to streamline the process of adding new users across multiple nodes, requiring only a single command for execution without necessitating the addition of an agent on each individual node.

To further refine user management, we implemented policies for user deactivation. Users inactive for over a year are automatically flagged for deactivation, aligning our user ecosystem with current operational needs and security protocols. This proactive approach ensures a dynamic and secure user environment across our systems. Instead of overly complicated software installations for users that likely needed a system administrator and further knowledge of accessing systems through SSH, our users could simply hit a web page, authenticate using their CACs, and get directly to the Open OnDemand page without any additional hurdles.

3.1.1.3 Slurm

As the HPCC underwent positive transformations, we saw an influx of users logging onto compute nodes and engaging in multiple job submissions. We use the term "submit jobs" in a loose way; a resource manager that restricts users from directly logging in to a compute node and monopolizing the remaining available resources was an alien concept. The ensuing resource conflicts led to user dissatisfaction, highlighting the need for a mechanism to impartially arbitrate and prioritize job submissions.

In response, we identified Torque/PBS and Slurm as industry-leading resource managers, testing both to determine their suitability. While either option was viable, we opted for Slurm due to its perceived ease of management from a system admin perspective. Slurm, an open-source, fault-

tolerant, and highly scalable cluster management and job scheduling system for Linux clusters, was chosen for its versatility.

Deploying the latest version from SchedMD introduced multiple bug fixes and additional features not available in the earlier version hosted on Red Hat's main repositories. This upgrade, coupled with network storage, paved the way for the first NIWC heterogeneous cluster, incorporating central processing unit (CPU) nodes, large memory nodes, and graphics processing unit (GPU) nodes.

Anticipating an increase in user numbers, the introduction of Slurm did not immediately translate into user satisfaction. The center users faced a learning curve in adapting to the new environment, despite efforts such as an HPCC Tutorials Hub page. Recognizing the need for a more user-friendly approach, Open OnDemand solved this final puzzle piece in the form of a browser-based job submission platform to conceal the complexities of Slurm, providing a more accessible experience for our users.

3.1.2 Operating System (OS) Standardization

A primary focus of the team was to enhance system security to prevent frequent quarantines. Recognizing the repetitive nature of this task, it was deduced that standardizing the OS across all servers would streamline the process. While Ubuntu and other Linux flavors lacked specific or mature Defense Information Systems Agency (DISA) STIGs and Red Hat Enterprise License (RHEL) 6 was nearing its end of life, RHEL 7 emerged as the optimal choice. DISA's well-established STIGs for RHEL 7, coupled with the command enterprise license agreement with Red Hat and widespread deployment at the enterprise level, influenced our decision. This OS standardization significantly reduced the administrative effort as we no longer had to navigate the intricacies of multiple OSs. We eventually upgraded the OS on all systems to RHEL 8 for the same reasons that we initially decided on RHEL 7 in 2019.

3.1.3 Ansible

Following the re-imaging of all servers to an upgraded RHEL version and their fortification through bash scripts and standardized configuration files, we aimed to integrate the incoming nodes into the network in the subsequent phase. To expedite this crucial process, we adopted Ansible as the scripting platform for orchestrating OS provisioning and server hardening. Ansible is an idempotent, agentless, flexible, secure, and elegantly simple Red Hat-supported application.

Operating on the principle of defining the end state of the system, Ansible automated node deployments and facilitated seamless reversion to a baseline state. Ansible playbooks streamlined the deployment of essential applications like MATLAB and Anaconda across each freshly inducted compute node. We used Ansible playbooks to enable the remote configuration and installation of Nagios Remote Plugin Executor on every node within the laboratory.

3.1.4 Nagios Core

Nagios provides monitoring and alerting services for pivotal IT components, encompassing servers, switches, applications, and services. Nagios plugins allowed us to generate comprehensive general availability reports and specific service check reports. Furthermore, Nagios monitors an array of parameters, including CPU workloads, server statuses, log activities, disk space utilization, core temperatures, and more. This concerted effort elevated the troubleshooting capabilities of the staff, offered greater visibility to the HPCC infrastructure, and served as a bulwark against downtime and potential work stoppages experienced by HPCC users.

4. SYSTEM HARDWARE AND VIRTUALIZATION

4.1 STORAGE

Following the consolidation of all HPCC servers into a single OS and the hardening of the OS to adhere to DISA STIG requirements, the next imperative task was addressing the fragmented storage infrastructure within the HPCC. The challenge at hand involved a swift transformation of the Hewlett-Packard Enterprise Modular Storage Array (MSA) 2050, originally configured as direct attachment storage (DAS) linked to a main server, into a network storage solution serving all HPCC compute nodes. While the DAS configuration exhibited commendable performance on the main server, extending its benefits to other servers was impossible under the current DAS deployment. Furthermore, transitioning from segmented to shared home directories in the HPC environment presented many advantages. The previous setup, with isolated home directories on individual nodes, created data silos and hindered collaboration.

4.1.1 Storage Area Network (SAN)

The decision to repurpose the MSA as a SAN allowed for block-level access to data, treating storage as a pool of blocks directly accessible by servers. This SAN adaptation proved beneficial for applications necessitating swift and efficient access to raw storage, including databases and virtual machine storage. The MSA was swiftly repurposed into a SAN, unifying all user storage into a singular home directory accessible across all compute nodes within the HPCC. However, the existing 8 Gigabit per second fiber channel and switches limited our progress. After a 6-month period, we quickly realized that the SAN storage was not the best fit for our environment and imposed additional maintenance burdens on the small team, including:

- Pacemaker administration and troubleshooting cluster problems.
- Monitoring and performance tuning.
- Documentation and configuration management.
- Training and skill development.

4.1.2 Network File Storage (NFS)

Confronted with the escalating costs of SAN maintenance, we promptly initiated an exploration of alternative solutions to provide our users with a consolidated storage platform. The subsequent logical progression led us to NFS, which assumes a pivotal role in HPC environments, presenting a distributed file system that yields numerous advantages. A primary strength lies in its ability to facilitate seamless file sharing and access across diverse nodes within a cluster, foster collaborative endeavors, and optimize resource utilization. NFS contributes to enhanced data mobility by empowering users to interact with files across different nodes in the HPC environment, thereby promoting a more adaptable and dynamic workflow. Furthermore, NFS streamlines data management by consolidating storage resources, simplifying maintenance tasks, and facilitating efficient data backup. This centralized approach not only bolsters system reliability but also mitigates the risk of data inconsistencies. In essence, NFS elevates the efficiency, flexibility, and reliability of data management in HPC, thereby supporting the seamless execution of intricate computational tasks.

4.1.3 Parallel File System

With the widespread adoption of GPUs in the HPC landscape and the rise of large language models (LLM), our attention has shifted to finding a more customized storage file system tailored for

AI and LLM applications. To meet the ever-growing demand for data in these domains, we have embraced the Lustre parallel file system, combining non-volatile memory express (NVMe) solid state drives (SSD) for high-performance hot storage with spinning disks for efficient cold storage. Lustre, designed to cater to large-scale, high-performance computing environments, enables parallel data access, making it ideal for AI applications dealing with extensive datasets and parallel processing requirements. Our recently acquired storage appliance seamlessly integrates NVMe SSDs and spinning disk technologies with hot-swap capabilities. The rapid NVMe SSDs offer a cost-effective, scalable, and efficient storage caching solution for AI workloads, while the spinning disks cater to the long-term storage requirements of cold data.

4.2 NETWORK

A fast network is of paramount importance in HPC clusters for several reasons, contributing significantly to the overall efficiency and performance of the computational environment. Additionally, a fast network is crucial for AI applications, with performance directly impacting the efficiency and speed of data transfer, communication between nodes, and overall system throughput.

Facing an outdated and slow network infrastructure was one of the main driving factors for upgrading the segmented network at HPCC, as both Ethernet and InfiniBand technologies were used.

- InfiniBand
    - InfiniBand is a high-performance, low-latency interconnect technology designed for connecting servers, storage systems, and network devices within HPC and data center environments. It is known for its high data transfer rates, low latency, and scalability.
- Ethernet
    - Ethernet is a widely used family of networking technologies that defines a set of standards for the physical and data link layers of communication in computer networks. Over the years, Ethernet has evolved to support higher data rates, improved efficiency, and various applications.

The decision was made to utilize Ethernet for a couple compelling reasons:

- Standardization and Widespread Adoption
    - Ethernet is a standardized and universally adopted networking technology, serving as the default choice for most networks.
- Familiarity and Expertise
    - The familiarity of Ethernet among IT professionals, coupled with a substantial pool of knowledge and expertise, simplifies the management of Ethernet networks, streamlining troubleshooting and maintenance tasks.
- Seamless Integration
    - Ethernet seamlessly integrates with existing IT infrastructure, facilitating a smooth integration process without significant disruptions.

Our choice resulted in the rapid deployment of a 100 Gigabit per second (100G) network infrastructure. We replaced the primary switch with a high-performance 100G switch and bolstered network capabilities by integrating Mellanox 100G network cards. Furthermore, we set up a dedicated 1 gigabit per second (1G) management network, addressing a vital void in the HPC cluster. This supplementary 1G network performs various roles, such as Nagios monitoring, baseboard management controller (BMC) operations, and streamlined access to the Tang server for efficient encryption key management.

4.3 VIRTUALIZATION AND INTERCONNECTIVITY

As the demand for HPC grows due to newer methods of modeling and the development of AI and machine learning (ML), the efficient and timely allocation and utilization of resources become critical. Introducing new technologies and design methods for HPC system architectures is crucial. This approach ensures that we not only meet but also anticipate the challenges of emerging workloads with complex models and larger datasets, maintaining agility to quickly address arising needs. Two key categories for the advancement of HPC systems architecture are software and hardware, specifically hardware virtualization and the interconnection of devices.

4.3.1 Server Virtualization

Virtualization is a technology that supports the division of resources across multiple logical devices known as virtual machines (VM) or containers to run on a single physical server (hypervisor) and performs a critical role in supporting HPC workloads, especially in environments where workloads are dynamic and unpredictable. Virtualization abstracts physical hardware resources, such as CPU, GPU, memory, storage, and network, allowing them to be presented as unique entities, ensuring that applications and OSs running on a VM do not interfere with each other. Virtualization pools and shares physical resources, enabling more efficient utilization and flexibility in resource allocation to each VM, and abstracts the underlying hardware, providing a layer of compatibility that allows VMs to run on different configurations of servers regardless of generation or underlying technology. The hypervisor is the core component of virtualization, managing, allocating, and monitoring resources for multiple VMs on the host system (server).

Virtualization boosts HPC by offering resource pooling and agile reconfiguration of resources on demand. This enhances the performance and efficiency of underlying resources, supports multi-tenancy, and simplifies operational management. The hypervisor and management systems provide the basis for resource consolidation, scalability, workload isolation, fault tolerance, and workload migration capabilities beyond what is typically achievable by “bare metal” OS deployments. Virtualization’s adaptability and elasticity make it the ideal solution for managing the demanding computational needs of HPC workloads while providing additional management tools to support improved system uptime and reduce time to deployment of new software or capabilities necessary to rapidly deploy the “high-speed, low drag” prototype initiatives seen in NIWC Pacific.

While some HPC environments are static in their use, others can involve running diverse and potentially conflicting workloads, often requiring a variety of network and software configurations. Virtualization platforms can host different OSs isolated by each VM instance, its software, and network configurations, enabling diverse applications within a single infrastructure. This flexibility simplifies resource management and maintenance, as different VMs can be customized to each specific task or software requirement while offering workload and data isolation, ensuring that one VM's activities do not interfere with another while protecting potentially sensitive data. This isolation enhances system stability and security by preventing resource contention and reducing data exposure to unauthorized users. Workloads can often demand substantial computing resources at a spectrum of scales and across a range of workload types, such as MPI, OpenMP, CUDA, and others. Virtualization consolidates resources by running multiple VMs on a single physical server, thus reducing underutilization. It provides machine-level characteristics with hypervisor-level awareness of underlying physical resources, such as NUMA and hyper-threading, often overlooked in custom HPC applications. Dynamic resource allocation via the virtualization hypervisor across host clusters ensures that VMs can adapt to workload fluctuations, optimizing resource usage and minimizing cluster hot spots when multiple workloads are executing simultaneously.

HPC environments fundamentally need scalability as a core characteristic, as the need for computational power can change rapidly with newer use cases. Virtualization provides a scalable and elastic infrastructure for HPC workloads, especially when combined with cloud computing, which can enable the rapid provisioning of massive pools of resources not typical of typical on-premises systems. Administrators can use software application passing interfaces (API) to integrate tools such as Slurm, a common job scheduler for HPC, with the virtualization controller to create, clone, or destroy VMs, allowing for dynamic scaling as computing demands fluctuate across tenants. This adaptability reduces the need for extensive physical infrastructure upgrades. It supports data isolation and on-demand resource allocation per workload while accommodating heterogeneous workloads on the same compute, network, and storage resources. Virtualization platforms offer advanced resource scheduling and load balancing mechanisms that operate independently of the hosted OSs, reducing the need to create complex configurations for the operating systems on which they are hosting critical applications. HPC administrators can allocate CPU, GPU, memory, and storage resources, ensuring that critical workloads receive the necessary resources without over-provisioning while also having the ability to add resources to live VMs with no reboot and while the system is under load. Load balancing algorithms distribute workloads across server clusters, optimizing performance by preventing overloading and underutilization of individual nodes. They adapt to workload shifts in real time based on performance data.

Beyond the core functions virtualization provides, it can also enhance the management and operational readiness of HPC systems by enabling support for routine maintenance activities and system resilience. Virtualization enables the creation of snapshots and cloning of VMs, facilitating backup, recovery, and rapid deployment of new instances of virtualized OSs and applications. They also support live migration in many circumstances, enabling the movement of VMs from one physical server to another without disrupting ongoing operations. Live migration, however, can be limited or unsupported if specific direct-to-hardware configurations are present. In those cases, however, offline migration is still generally supported and continues to present an advantage in cases where the migration of the software is paramount (e.g., hardware upgrades). Additionally, features like fault tolerance clustering further enhance system resilience and reliability, where 99.99%+ uptimes are desired.

4.3.2 Device Interconnection

As much as virtualization technologies have advanced in their capabilities to support modern workloads in high-performance computing environments, server hardware likewise continuously improves to provide new capabilities that support faster, more efficient processing of data. These improvements, however, now differ from those of the past, when the application of Moore's law had a profound impact on ever-increasing computing capacity. Thanks to generational leaps in CPU capacity, today's advancements are found in the improvements to the interconnectivity of resources and devices across the local system bus. Two emerging technologies, compute express link (CXL) and cable-based peripheral component interconnect express (PCIe), provide an early look at how server platforms may make significant changes to interconnection and communication across a system.

The CXL, an open standard-based, industry-supported interconnect, offers cache coherence for processors, memory expansion, and accelerators. It is designed to meet the growing demand for higher performance and bandwidth in modern HPC environments. CXL is built on serial PCIe interfaces and aims to provide a high-performance and scalable interface for connecting various accelerators, memory devices, and CPUs. CXL supports cache coherency between processors and attached accelerators, ensuring that the data remains consistent across different components of the

system. It also allows for memory expansion by enabling the attachment of memory devices (such as persistent memory or high-bandwidth memory) directly to the CXL fabric.

One of the primary applications of CXL in HPC is the connection of accelerators like GPUs and field-programmable gate arrays (FPGA). CXL enables direct communication between the CPU and accelerators, facilitating efficient data transfer and processing. CXL facilitates the integration of diverse accelerators and memory types within an HPC system. This enables researchers and engineers to build more flexible and powerful computing architectures tailored to data-centric workloads, where efficient data movement and processing are critical. This includes applications in areas such as scientific simulations, AI, ML, and data analytics.

PCIe is a well-established high-speed serial computer expansion bus standard commonly used for connecting various hardware devices, such as storage controllers, expansion cards (i.e., networks, GPUs), and other peripherals, to the motherboard of a computer. The PCIe protocol offers advantages not provided by other fabrics, including low latency and guaranteed transmission with power-saving features. The introduction of PCIe over cable enables innovative HPC architectures by leveraging copper or fiber optic cables as the transmission medium. Traditional systems, which place expansion cards and controllers near the CPU inside the server chassis for optimized performance, often face issues with server form factor and increased power and cooling needs. In contrast, PCIe over Cable technologies use the high bandwidth and low latency of the PCIe protocol at distances up to 100 meters. PCIe over cable input/output (I/O) expansion connects a server to one or many PCIe devices. It allows the offloading of dedicated expansion cards to external chassis, offering system designers greater flexibility in configuring hardware assets to meet best practices for each device type or non-normative requirements.

Switching devices for PCIe over cable are crucial in creating a scalable PCIe fabric solution. They operate over networked backplanes using standardized external cables, typically copper or fiber-based. These switches facilitate efficient communication between various components within an HPC system, including CPUs, GPUs, FPGAs, accelerators, storage, and memory devices. Switches are able to support both transparent and non-transparent bridging use cases for either I/O expansion or clustering applications, while larger-scale configurations can be realized by interconnecting multiple switches. Additional configurations supporting multiple hosts, switch partitioning, and fail-over configurations are supported in varying degrees across the available vendor platforms. Management and configuration of the PCIe fabric occur through the Ethernet port. Most vendors support proprietary APIs, software development kits, drivers, and GUI interfaces, enabling tailored configuration and integration for each HPC environment. Software drivers and APIs enable the dynamic allocation and re-allocation of I/O devices to servers, including the ability to hot-add or remove expansion devices. This facilitates near-real-time reconfiguration of server hardware to meet specific workloads or introduce new capabilities without requiring a complete redesign of the fundamental supporting hardware.

CXL and PCIe over cable present notable advancements in HPC environments. They integrate diverse expansion devices, like FPGAs and GPUs, into unified, high-speed interconnect fabrics, enabling efficient, high-speed data sharing. With the ability to provide high-bandwidth, low-latency connections, they facilitate the creation of more powerful and scalable HPC systems while enhancing system flexibility and resource utilization, enabling scientists to research complex computational challenges with improved speed and efficiency. By dynamically allocating expansion devices as needed, HPC environments can rapidly reconfigure to changing workloads and, combined with the virtualization of software, ensure that each application and tenant receives the necessary resources without costly idle time or underutilization.

This provides a greater level of efficiency, reduced resource contention, and a more responsive, scalable HPC infrastructure capable of handling diverse computational workloads in a more cost-effective and manageable manner compared to traditional solutions.

5. CONCLUSIONS

Building an enterprise-level system requires thought and research, a willing, tenacious team, and vision; however, within all of those facets, it requires flexibility in design and execution. Technology can shift rapidly, rendering what seemed like a good idea at the time completely obsolete the next day, so it is paramount when designing systems to consider longevity through flexibility and make sure systems can withstand waves of technological evolution. Our experiences captured in this paper do not present a step-by-step guide, as that would likely be dated by the time it was published; rather, it looks at the ways in which we conducted business to maintain a forward-facing position to support the projects working critical efforts at our command and have the preparation to adjust to new requirements as they came in. Further, we looked at ways to lower the administrative burden on our personnel to provide space for the entire team to consider future implications in design. This prevented the team from only focusing on the here and now, a trap that many can fall into easily, which may satisfy the current needs but exponentially raises the difficulty of managing change and the needs of a diverse set of requirements.

A key takeaway from this endeavor of bringing our systems from a complicated, nearly unusable collection of separated systems into the current state was to maintain communications with multiple groups across the command and actively seek out efforts to understand their needs. Nearly half of the groups currently using our systems previously didn't know we existed or simply didn't think that we would be useful. It was imperative that we first understood their efforts so that we could present what we could offer them and, further, how we could tailor our systems to be useful to a broader collection of personnel. By continuing this methodology, our systems organically build themselves to the users' needs; we get more users interested in and using our systems; we interview them; and the cycle repeats for an ever-expanding, improving SoS that operates successfully at the enterprise level.

This page is intentionally blank.

INITIAL DISTRIBUTION

84310	Technical Library/Archives	(1)
53236	S. Pease	(1)
53236	G. Istambouli	(1)
53236	Z. Haddad	(1)
53236	D. Arevalo	(1)
53633	D. Phillippi	(1)

Defense Technical Information Center
Fort Belvoir, VA 22060–6218 (1)

This page is intentionally blank.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-01-0188

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
January 2024	Final	

4. TITLE AND SUBTITLE

Building the Evolving and Flexible High Performance Computing Enterprise

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHORS

Stephen T. Pease
George Z. Istambouli
Ziad A. Haddad
NIWC Pacific

Daniel J. Arevalo
Daniel W. Phillippi
NIWC Pacific

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

NIWC Pacific
53560 Hull Street
San Diego, CA 92152–5001

8. PERFORMING ORGANIZATION REPORT NUMBER

TD-3430

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release. Distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This paper explores a brief discussion of the successes and challenges faced when building an enterprise-level high-performance computing center that serves an entire command. Methods for achieving systems that are broadly accessible, easily usable, and still flexible enough to meet the evolving needs of numerous technology areas are explored and highlighted. Though not a step-by-step guide, this paper seeks to present a pathway to achieving a malleable environment to serve numerous competing priorities across a command by demonstrating results and sharing the experience of a successful build at the Naval Information Warfare Center Pacific.

15. SUBJECT TERMS

High Performance Computing; High Performance Computing Center; Enterprise Systems; Cluster Design; Open OnDemand

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Stephen T. Pease
U	U	U	SAR	32	**19b. TELEPHONE NUMBER** *(Include area code)* (619) 221-5698

This page is intentionally blank.

This page is intentionally blank.

Approved for public release. Distribution is unlimited.

Naval Information Warfare Center (NIWC) Pacific
San Diego, CA 92152-5001