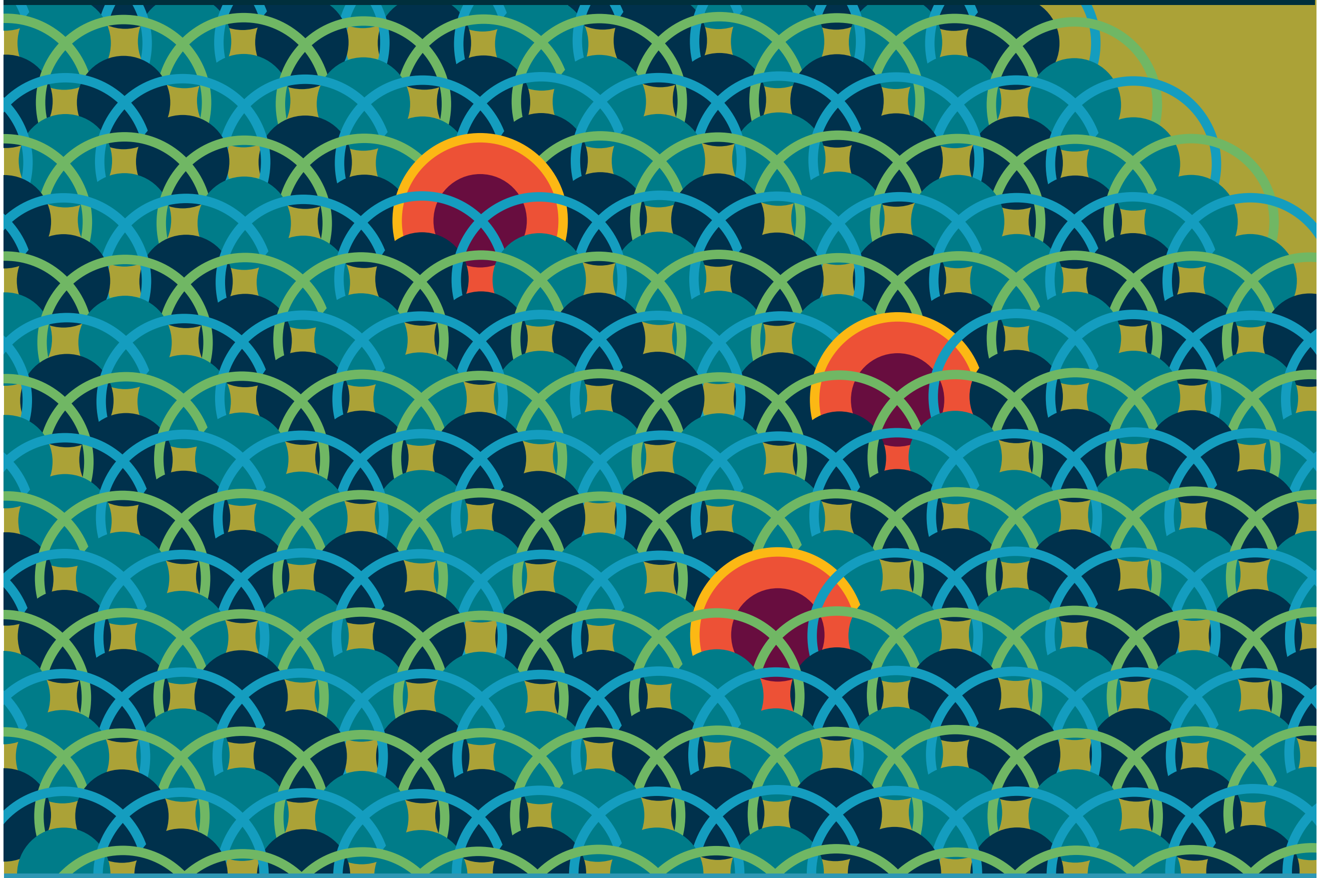
ASCR Cybersecurity for Scientific Computing Integrity



DOE Workshop Report

January 7-9, 2015
Rockville, MD

U.S. DEPARTMENT OF ENERGY
Office of Science

BERKELEY LAB

ASCR Cybersecurity for Scientific Computing Integrity

DOE Workshop Report

January 7–9, 2015
Rockville, MD

LBNL-6953E (2015)

Workshop Organizing Committee

Sean Peisert (Co-Chair), Lawrence Berkeley National Laboratory and University of California, Davis

George Cybenko (Co-Chair), Dartmouth College

Sushil Jajodia (Co-Chair), George Mason University

David L. Brown, Lawrence Berkeley National Laboratory

Christopher L. DeMarco, University of Wisconsin-Madison

Paul Hovland, Argonne National Laboratory

Sven Leyffer, Argonne National Laboratory

Celeste Matarazzo, Lawrence Livermore National Laboratory

Stacy Prowell, Oak Ridge National Laboratory

Brian Tierney, Energy Sciences Network (ESnet)

Von Welch, Indiana University

DOE ASCR Point of Contact

Robinson Pino

Cover: A key element of scientific computing integrity is finding small anomalies in larger patterns. Whether introduced accidentally or maliciously, such anomalies can reflect behaviors that undermine the integrity of the scientific results and can have far-reaching implications.

Contents

Preface

At the request of the U.S. Department of Energy's (DOE) Advanced Scientific Computing Research (ASCR) program, a workshop was held January 7–9, 2015, in Rockville, Md., to examine computer security research gaps and approaches for assuring scientific computing integrity specific to the mission of the DOE Office of Science. Issues included research computation and simulation that takes place on ASCR computing facilities and networks, as well as network-connected scientific instruments, such as those run by other DOE Office of Science programs. Workshop participants included researchers and operational staff from DOE national laboratories, as well as academic researchers and industry experts. Participants were selected based on the prior submission of abstracts relating to the topic. Additional input came from previous DOE workshop reports [DOE08,BB09] relating to security. Several observers from DOE and the National Science Foundation also attended.

The workshop was divided into four topic areas: **1 Extreme Scale Power Grid Simulation**, **2 Trustworthy Supercomputing**, **3 Trust within High-end Networking and Data Centers**, and **4 Extreme-Scale Data, Knowledge, and Analytics for Understanding and Improving Cybersecurity**. Participants were divided into four corresponding teams based on the category of their abstracts. The workshop began with a series of talks from the program manager and workshop chairs, followed by the leaders for each of the four topics. The rest of the workshop consisted of topical breakout discussions and focused writing periods that produced most of this report. Although the workshop was organized around four topics, this report is structured around the latter three because they focus more clearly on the primary issue of security of scientific computing integrity, rather than computer security more broadly. However, some of the text about *Extreme Scale Power Grid Simulation* remains in this report as a motivating example of the need for ensuring scientific computing integrity.

Executive Summary

The Department of Energy (DOE) has the responsibility to address the energy, environmental, and nuclear security challenges that face our nation. Much of DOE's enterprise involves distributed, collaborative teams; a significant fraction involves "open science," which depends on multi-institutional, often international collaborations that must access or share significant amounts of information between institutions and over networks around the world. The mission of the Office of Science is the delivery of scientific discoveries and major scientific tools to transform our understanding of nature and to advance the energy, economic, and national security of the United States. The ability of DOE to execute its responsibilities depends critically on its ability to assure the integrity and availability of scientific facilities and computer systems, and of the scientific, engineering, and operational software and data that support its mission.

The large-scale science and energy research funded by DOE increasingly relies on large-scale computational modeling and simulations, as well as on capturing data from scientific instruments, and then analyzing, transmitting, storing, and sharing that data all within computational environments. Much of that research has results that are purely scientific, while some of the research findings, including those from computational results, can also inform national policy decisions. Moreover, the areas for which DOE is uniquely responsible, including energy, environment, and nuclear weapons, all directly affect our nation's future security and prosperity. And in each case, scientific computing integrity assurance is extremely important. Even for the basic science, since U.S. taxpayer dollars fund a large cadre of the nation's top scientists to do research, it is vital that the results can ultimately be trusted. For applied science, the integrity of the computations and the data used to achieve these results is critical to provide confidence in any resulting policy decisions, as well as ensuring the safety of DOE's own scientific instrumentation infrastructure. However, even when simply considering investments within the DOE itself, it should be noted that computational simulations are increasingly used in the design and operation of advanced DOE user facilities, representing a considerable investment of public funds. Thus, even at this level, it is imperative that computational simulation results be trustworthy to avoid waste and misuse as a result of the policy decisions to invest in such facilities.

We define scientific computing integrity as the ability to have high confidence that the scientific data that is generated, processed, stored, or transmitted by computers and computer-connected devices has a process, provenance, and correctness that is understood. Vital components of scientific computing integrity are also metrics and measures of both integrity and uncertainty in order to evaluate how much confidence can be placed in that data. Thus the development of advanced scientific computing methodologies for the design and evaluation of security of large-scale computational systems in the interests of assuring scientific computing integrity is of vital importance. DOE science relies on both commodity and exotic technologies, including software, data, and hardware computing assets that have risk profiles that are poorly understood by the research and computer security communities. Even when DOE science uses commercial off-the-shelf (COTS) computing infrastructure, the science being supported has workflows often not seen elsewhere in the computing community, meaning that the consequences of security risks to scientific computing integrity are not well understood.

Research is needed into security techniques appropriate for open scientific environments. "Classical" computer security techniques work primarily by restricting access and limiting information flow. This is because many of the original techniques were developed to protect military systems, where high-assurance confidentiality and integrity are paramount. And this is still often true of modern security research results developed for the purposes of other U.S. government agencies such as the Department of Defense, the Department of Homeland Security, and the intelligence community. However, security strategies centered around highly restrictive access controls are often inappropriate in open scientific environments. (Indeed, as exemplified by the numerous security breaches involving large-scale data thefts in 2014, these techniques may be ineffective even in non-scientific environments.) Regardless, in open scientific environments where computational throughput is a primary goal, there is clearly a critical tradeoff between openness and classical computer security techniques that emphasize greater isolation. Thus, new research is needed to explore technologies in order to preserve and maximize the scientific openness necessary to DOE's scientific infrastructure while ensuring integrity of the science conducted using that infrastructure. Moreover, successful research in this area may well have applicability beyond DOE's mission space.

Vision and Goal. The vision and goal of this report is to identify fundamental research challenges to enable scientific computing integrity and computer security by achieving repeatable, reproducible workflows that produce computing results whose process, origin, and data provenance is understood, whose correctness is understood, and for which uncertainty estimates are provided. Accordingly, these capabilities must be enhanced by systems with autonomous decision-making capabilities responding at light speeds, giving scientists the ability to make informed decisions about the integrity of their data.

Measures of Success. Success in scientific computing integrity would ideally be to have provably secure extreme-scale computing systems and workflows. In the absence of provably secure systems, success would entail having extreme-scale systems with some provably secure components and reliable, useful data describing the events taking place in those systems, that, with the proper analytics, can accurately characterize security-related events that affect scientific computing integrity.

Research Recommendations

As we discuss later in the report, several key research strategies to achieve this success include:

Enhance the "trustworthiness" of DOE supercomputers by developing:

- means to build solutions for assuring scientific computing into the design of supercomputers;
- robust means for evaluating ways in which a system composed of interconnected, networked elements can affect scientific computing integrity;
- precise and robust means of capturing the right data to provide concrete evidence of scientific computing integrity such that reproducibility is possible and also so that integrity can be verified when it is maintained or diagnosed when it cannot;
- metrics for quantifying the trustworthiness of scientific data, capturing the likelihood and potential magnitude of errors due to uncertain inputs, incomplete models, incorrect implementations, silent hardware errors, and malicious tampering; and
- significantly improved means for balancing the assurance of scientific computing integrity between hardware and software to best monitor and maintain integrity while also minimally impacting the throughput of scientific research.

Develop means to assure trust within open, high-end networking and data centers by performing research to:

- understand the resilience of DOE scientific computing to integrity failures in order to understand how to best create data centers to support increasing computing integrity;
- explore how the evolution of virtualization, containerization, and modular runtime environments impact scientific computing integrity, and where control, layering, and modularity enhance integrity assurance, and where it adds complexity and scaling problems;
- understand how to create new, scalable techniques that enable the secure tagging of data and network packets in real-time for subsequent policy management and forensic analysis; and
- create means for developing coherent authorization and access controls particular to the open science mission, which can maximize integrity and computing efficiency.

Research and develop means to collect extreme-scale data and knowledge, and develop and apply analytics in order to understand and improve scientific computing integrity and computer security by:

- developing an analysis framework capable of collecting scientific computing integrity data at an unprecedented scale from multiple sources that collectively represent the system under study to enable adaptive, streaming analysis for monitoring and maintaining scientific computing integrity;
- developing means to learning and maintaining interdependent causal models of the scientific computation, exascale system, and computer security in real-time to enable better, faster recovery to reduce disruptions to scientists' efforts;
- developing capabilities to model, quantify, and manage exascale performance to allow exascale computing users and system operators to effectively manage the tradeoffs between scientific throughput and scientific computing integrity performance; and
- develop new methods for meaningful risk measures and threat measures of HPC integrity.

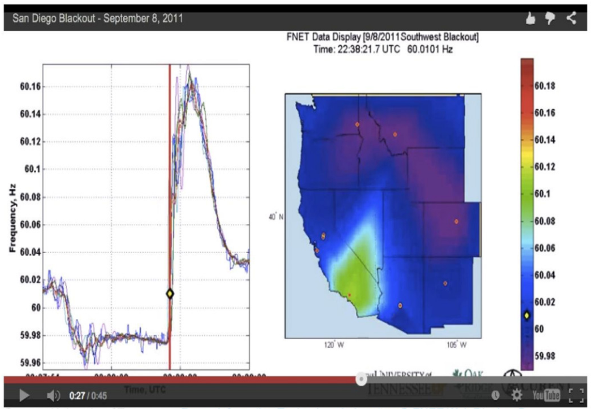
No research program currently exists within DOE or elsewhere whose mission is to produce research results that will allow these objectives to be achieved. A new program in this area must leverage the current strengths within ASCR's Applied Mathematics program in predictive modeling and simulation and data analysis, as well as strengths within the Computer Science and Next Generation Networking for Science programs for developing trustworthy supercomputing and high-end, trustworthy networking systems. Additionally, this research effort should have strong ties to exascale efforts. Notably this should include the aspect of the resilience effort focused on fault detection—a program in scientific computing integrity could extend that work such that when a fault is detected, research results may help to correlate with parts of the system to ensure that the fault is not caused due to malicious intent. This research effort should also have ties to the X-stack effort, which is also focused on co-design of hardware and software suitable for exascale systems. Building security into that stack from the outset is vital to scientific computing integrity. Finally, this effort should have close ties to Office of Science facilities, including both traditional computational and networking facilities such as ESnet, NERSC, and the Leadership Computing Facilities, but also "cyber-physical" scientific instruments such as the light sources and particle accelerators.

1 Introduction: A Research Path for Assuring Scientific Computing Integrity

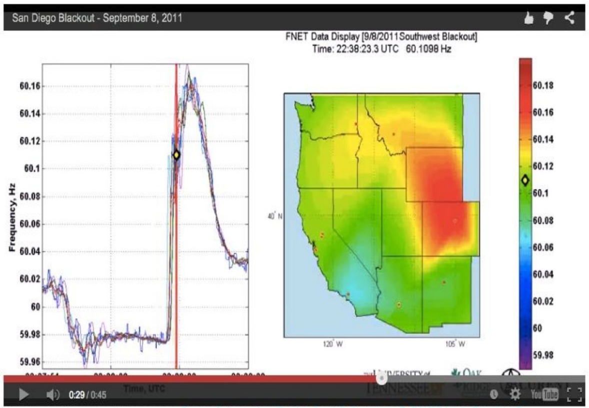
DOE has the responsibility to address the energy, environmental, and nuclear security challenges that face our nation. Much of the department's enterprise involves distributed, collaborative teams; a significant fraction involves "open science," which depends on multi-institutional, often international collaborations that must access or share significant amounts of information between institutions and over networks around the world. The mission of the Office of Science is the delivery of scientific discoveries and major scientific tools to transform our understanding of nature and to advance the energy, economic, and national security of the United States. The ability of the department to execute its responsibilities depends critically on its ability to assure the integrity and availability of scientific facilities and computer systems, and of the scientific, engineering, and operational software and data that support its mission.

The ability to assure that integrity and availability of scientific facilities, computer systems, and data is a monumental and very difficult challenge. However, given the critical impact of the scientific results of DOE research on the nation's well-being, developing new means to assure scientific computing integrity is vital.

High-Consequence Examples. Computing increasingly plays a critical role in many areas of modern science. However, that science often also plays a role in public policy, economics, and infrastructure development. As such, were some silent failure of scientific computing integrity to occur, the effects could have far-reaching consequences. For example, scientific computing related to energy and climate research could lead to the insertion of flaws in the design of our energy infrastructure, or the generation of incorrect data on which public policy relating to climate and energy policy is made. In both of these cases, the DOE relies on extreme-scale computing in order to perform much of the research analysis. And, were the scientific computing integrity of power grid or climate research to fail in a silent way, policy changes relating to energy infrastructure design and energy consumption in the United States might be put into effect that could make the power grid less stable, or lead to improper responses to current grid stability issues. To illustrate the potential cost of grid instability, consider the 2003 blackout which began with a single-line failure in Ohio and spread to the Eastern seaboard, ultimately affecting 50 million people and inflicting costs estimated at up to $10 billion. And the risk of such a failure is real—the power grid has been described as the "most complex machine ever built [Ami02]," and it is this complexity that can enable integrity failures to be either accidentally masked or intentionally hidden. Indeed, the size and complexity of power grid modeling and simulation is an HPC grand-challenge problem in its own right [MMCS11]. That said, while extreme-scale computing is an important component of power grid research, it does not, however, stand on its own. The power grid comprises a massive number of control systems. To include actual control systems in any computational power grid analysis, extreme-scale, trustworthy cyber-physical network testbeds must also be in place.



Source: University of Tennessee, Power Information Technology Laboratory, FNET Web Display.
Note: Normal system frequency of 60 cycles per second appear as blue. Green and red colors indicate higher than normal frequency.



Source: University of Tennessee, Power Information Technology Laboratory, FNET Web Display.
Note: Normal system frequency of 60 cycles per second appear as blue. Green and red colors indicate higher than normal frequency.

Figure 1: Illustration showing grid disturbances, including the scale and speed of the effect of the disturbances. These two maps show the propagating frequency disturbance at time points separated by less than two seconds.

In addition to energy grid and climate modeling, there are numerous other areas of scientific computing that have significant practical and policy impact. Other examples of potentially dangerous results due to loss of scientific computing integrity include:

- flawed genomic research and protein analysis of biological energy sources could lead to overstating or understating the impact of renewable energy sources
- erroneous results in computational seismic research and simulation, affecting building designs and undermining protection for people and property
- erroneous computational results for material property analysis, leading to flawed material production and use in products ranging from computers to automobiles to aircraft.

Then there is the wider issue of overall protection of systems and information. Recent news stories have illustrated the breadth of the challenge facing the nation and world:

- The world's largest personal computer maker had installed software on its computers that monitored users' activity without their knowledge and could have been used by third parties to breach security systems.
- Major firms' databases are routinely hacked, resulting in the personal medical and financial information for tens of millions of individuals falling into the wrong hands.
- A leading entertainment company's systems were hacked, leading to the release of information embarrassing to the company, employees' personal data, and the threat of further action unless the company altered its business plan.
- In several regional conflicts in Eastern Europe, the network infrastructure was an early target of the aggressors.

Finally, activity by employees inside institutions can also pose a threat, whether by releasing sensitive information about political activities and intelligence strategies, or using restricted systems to design military systems for a foreign nation, as has happened in the past.

1.1 Ensuring Scientific Computing Integrity is Different from Traditional Computer Security

The "C-I-A" triad of goals for computer security—confidentiality, integrity, and availability, has guided most system development for the past several decades. Confidentiality in particular has had a major role, given that much of the original computer security work derived from U.S. Department of Defense needs and funding. For example, early efforts in computer security focused on modeling access controls and limiting information flow [Bib77, BL73, Den76, GM82, HRU76]. However, the goals of the Office of Science are largely distinct from the Department of Defense (DoD), Department of Homeland Security (DHS), and even National Nuclear Security Administration (NNSA) foci, which are subject to different constraints and their own unique challenges, often centered around the notion that confidentiality needs are paramount. Indeed, solutions appropriate to such facilities might hurt open and international science where availability and data sharing are often of greatest importance.

Moreover, DOE Office of Science solutions are also very different from general purpose computing as well. For example, DOE's high-performance computing and large-scale science instrument workflows differ from those in general-purpose computing in that each individual workflow can require extremely high-performance and also highly distributed networking and computing infrastructure. This stands in contrast to general-purpose computing that might collectively require high-performance, distributed infrastructure (e.g., commercial video streaming services such as Netflix or YouTube) but for which the individual processes have comparatively miniscule resource requirements. Additionally, those scientific workflows are often much more well-defined and have use cases that vary less often than general-purpose computing. For example, DOE supercomputers might run one program that runs for days or weeks on tens of thousands of processors, and can often run a very small handful of scientific applications over and over again for months in a predictable way, depending on when a scientist submits a computation to the job queue. Consider this in contrast to someone working in an office environment who switches back and forth between their word processor, email program, calendar, contact manager, and web browser (with perhaps dozens of different sites visited) perhaps many times within a few minutes at intervals that are not easily predictable or consistent from one person to the next. In the latter situation, anomaly detection systems that label behavior that statistically deviates significantly from normal behavior as "malicious" frequently cannot be effective because the variability of programs and users is so high that malicious behavior is lost in the noise [SP10]. In contrast, the regularity of the workloads in high-performance computing environments provides an opportunity to implement stronger anomaly detection with a much lower error rate.

Regularity of behavior patterns on supercomputers versus conventional platforms is unique to DOE because of the development of HPC software within DOE and the requirements that it be performance portable and uniformly robust (debugged) between large- and small-scale computing architectures. One expects a set of CPU cores that process a given HPC workload to show a largely similar behavioral pattern, as has been successfully demonstrated in past efforts, to "fingerprint" what is running on supercomputers and verify that it is within policy for what a user is supposed to be running on DOE resources [Pei10, SP10, WEPB12, WPB13]. This natural homogeneity is the friend of the defender as benign deviations of behavior are less likely to occur.

Specifically, the Office of Science must provide assurance for availability and integrity of facilities and data for open scientific research, including international collaborations and extremely data-intensive applications. The Office of Science labs and user facilities function in completely open environments and are often accessed by authorized users around the world. Thus, the primary goals of the Office of Science are to enable collaborations and open data sharing, unlike many other U.S. government agencies, for which the primary goals are often to strictly restrict access to all data on machines. Moreover, the security controls that are used must be minimally intrusive both for the scientific users and the computational environment that the scientific analysis is performed in. For example, on an Office of Science system, multiple research projects from multiple countries run on the same machine but some side channel attacks to determine other user actions may be tolerable. Additionally, discretionary (not mandatory) access control paradigms are used. They allow external network connections and do not use traditional "stateful" and or "deep-packet inspecting" firewalls due to the extreme, negative impact on network throughput. Thus, while compute facilities make reasonable efforts to protect data confidentiality, in comparison to many other environments they must accept more risk in the interest of their primary mission of advancing science goals. However, scientific computing integrity and availability still remain paramount goals.

Assuring the integrity of DOE open scientific processes requires an ability to look at heterogeneous sources of information, such as the network, computing nodes, scientific instruments, storage systems, operating systems, runtime and applications, and detect patterns that represent faults of various kinds, including incidental and intentional corruptions. Understanding the source of the faults, and in particular determining if these result from malicious attacks (including insider attacks), accidental failures, or natural faults (e.g., cosmic rays causing bit flips, drive failures, etc.), requires the ability to classify the faults. Presenting the results of such a failure analysis in a useful way to an end user is also essential, as is determining appropriate mitigation, remediation, and recovery strategies.

It is important to note that although on one level, it does not necessarily matter whether a fault is intentional or not—a failure to scientific computing integrity has occurred and must be corrected, regardless of whether a user accidentally or intentionally caused that fault—there are key distinctions that should be considered. For example, a benign or natural fault is more likely to be caught by existing fault-tolerance techniques, be they fault tolerance built into the computer system, or the "fault tolerance" of science itself in which multiple experiments are run repeatedly by multiple, independent teams to see if the same results are achieved. A natural fault or accidental failure may well show up as an inconsistency in such a case. In contrast, a motivated attacker could theoretically alter the results in a consistent way, thus potentially evading detection entirely despite the redundancy already built into computer systems and scientific processes. An insider threat—someone with increased levels of access to or knowledge of a system and/or trust by an organization [BEF+10]—would be particularly well-suited to creating such a consistent attack. The fact that motivated, malicious attacks are harder to detect does not necessarily make them greater risks, however. For example, the amount of scientific computing integrity failures today due to user error or natural fault is likely to be high, given the increasing complexity and scale of the computing systems being used. Thus, we believe that at present, non-malicious failures are at least as significant a problem to be addressed through additional research as malicious attacks. Exascale computing will increase this challenge even more as system architectures scale to many millions of processor cores.

From a DOE open science perspective, the complex systems that must be understood from a computing integrity and computer security perspective include

Office of Science scientific user facilities, including high performance computing environments such as NERSC and the Leadership Computing Facilities; extreme scale network facilities, notably ESnet; as well as numerous networked scientific instruments, such as light sources and particle accelerators.

Open DOE science represents a federated community of data sources (including data repositories and scientific instruments of varying scale), high performance computing and networking, computational resources, and an international community of researchers who generate vast quantities of research output from them. Additionally, data centers and networks combine to provide the mechanism by which data is stored, discovered, and transported. Additionally data centers support a number of the underlying mechanisms for computational science, including managing the authentication and authorization of scientists and providing interfaces used by scientists and others in the community for computation, visualization, data management, and other computational scientific analyses.

1.2 Challenges and Opportunities in Ensuring Scientific Computing Integrity

Computing system integrity—due to attacks against and vulnerabilities in public, private, academic, and commercial facilities—has come to the public forefront. Today, computer security is largely dependent on known threat models developed from within application domains where they have been particular foci, such as national security, financial security, or health privacy. Risks can then be modeled or categorized according to the "confidentiality, integrity, and availability (C-I-A)" methodology. However, DOE high-performance computing requirements are different than many of these traditional environments. In contrast to a DoD weapons system or financial transactions in a large banking form, potential scientific risk examples may include:

- numerical uncertainty or computational variance for experiments and simulations
- vulnerabilities of politically sensitive scientific data (e.g., climate data) to deliberate deception from political or financial actors
- vulnerability of networked-equipped DOE instruments and facilities (e.g., remotely controlled experiments) for misuse.

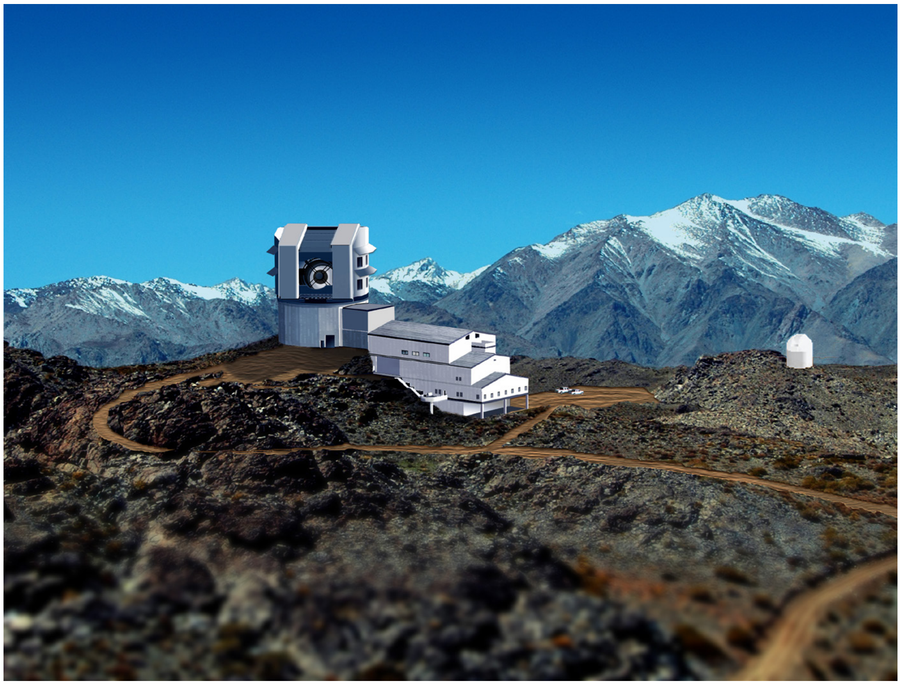
The DOE has a well-defined mission of encouraging and supporting the security of the open science institutions that it supports. This requires deep understanding of that mission and the dependencies that open science has on network-connected computer systems. For instance, attacks that might otherwise be considered mundane or uninteresting might collectively result in an erosion of trust in scientific results. Addressing this requires a study of important factors that might degrade the effectiveness or impact of open science, e.g., threats to the integrity of result reproducibility, experimental accuracy, etc. Moreover, it has become clear that detection and response are equally as important as a priori prevention of scientific computing integrity manipulation.

The increase in the sophistication of computational scientific workflows makes for a complex ecosystem, as authentication needs for multi-site activity strains the traditional notion of authentication and authorization. In addition to workflow complexity, data volumes are growing extraordinarily fast, as are the ongoing site and system architecture growth in global file systems, the integration of software defined networking (SDN), Internet Protocol version 6 (IPv6), and the march toward exascale computing and the looming computational challenges that go with it. To be clear, these changes bring both security challenges as well as opportunities for improved security [Mon13]. To address these challenges, it has become clear that a re-examination of foundational issues surrounding how we enable scientific computing integrity and computer security in large user-base open scientific instruments and their environments is needed.

Open DOE scientific facilities such as light sources and particle accelerators, and international scientific facilities used by DOE researchers, such as the

The Advanced Light Source provides 39 beamlines for studying a wide range of scientific problems.

Large Hadron Collider (LHC) and the Large Synoptic Survey Telescope (LSST) are reliant on fast, secure network connections. These connections primarily support the transfer of sensor measurement datasets. Some of these network connections must carry extremely large volumes of data to and from remote sites not co-located with storage and computing equipment, such as the LSST which sits at 2,682m above sea level on a mountain in Chile. This creates challenges both with regard to security and policy issues that arise on international networks, but also technical challenges relating to maintaining throughput of the data transfer, and consequently, integrity of the science. At the same time, network interfaces are being integrated directly into these instruments and associated devices, thereby making them potentially reachable from outside networks, thereby increasing their vulnerability. And, unlike many enterprise computing environments, computing infrastructure supporting large-scale scientific instruments is often "frozen" for very long science runs (weeks, months or years), creating a tension with typical computer security patching techniques.



The Large Synoptic Survey Telescope, being built on a remote mountaintop in Chile, is expected to start gathering data in 2019. Artist rendering by Michael Mullen.

HPC is an integral part of DOE's mission and the DOE labs have some of the most advanced supercomputers in the world. The DOE ASCR computing environment is unique in many ways:

- Sheer size of data: A full-configuration run on NERSC's "Edison" supercomputer can use almost 400 TB of RAM.
- Short lifetime of DRAM data: DOE applications update memory frequently.
- Number of processors in use: 10,000+ processor node runs are commonplace.
- Frequent tightly coupled communication: DOE applications communicate data between processors at high rates, requiring both short (microsecond) latencies and high bandwidth (10–100 Gbps) communication between nodes.

The Cray XC30 supercomputer Edison at NERSC.

Even small computing jobs (e.g., 1 percent of an HPC system's full configuration or a few hundred nodes) on a DOE supercomputer are large compared to typical server or database workloads. DOE applications use unique communications patterns such as collective MPI and parallel global address space to perform parallel processing (not client-server, and no notions of upstream or downstream network). The DOE supercomputing workload consists of scientific applications highly tuned for these specialized hardware environments. DOE has decades of experience running these applications addressing large problems at scale on these platforms. In a sense, DOE's computational needs have been met by vertically integrated hardware and software platforms, consisting of high-end processors, fast interconnects, optimized parallel runtimes, and tuned scientific applications.

Despite the challenges posed by the complexity of this environment, there is also opportunity. DOE's HPC systems run very special software stacks. DOE develops many of its own applications. DOE has a large investment in systems and numerical libraries that form the foundation of those applications. In procuring and deploying HPC systems for many years, DOE has already confronted severe performance imbalances and obstacles to scalability. Going forward, it should seek programming models that better adapt to both the performance and reliability of available resources to achieve correct process completion. No institutions are better positioned to understand this software, and be able to assure its integrity using well-defined automated processes, than the DOE national laboratories already developing the software and hardware stacks.

1.3 Toward a Path of Assuring Scientific Computing Integrity

Success in scientific computing integrity would ideally be to have provably secure extreme-scale computing systems and workflows. In the absence of provably secure systems, success would entail having extreme-scale systems with some provably secure components and reliable, useful audit log data that, with the proper analytics, can accurately characterize security-related events that relate to verification and validation of scientific computing integrity.

In an ideal world, it would be possible to construct a system by bootstrapping only from components whose behavior and trustworthiness can be can be proven or characterized from first principles [PTB12]. But even for specialized supercomputers, this is likely out of reach due to cost and scaling reasons. That said, formal verification is still an essential component of assuring scientific integrity, and indeed, several areas in computer science have already shown substantial success in formal verification. However, substantial research must still be undertaken to determine how to develop means for verifying supercomputing applications and environments, and to determine how the resulting techniques can be integrated into the development cycle. Therefore, **Section 2** of this report focuses on *Trusted Supercomputing*, largely centered around research into the co-design of hardware and software systems that can be assured in this manner as much as possible.

Section 3, *Trust within Open, High-End Networking and Data Centers*, is closely related to Trustworthy Supercomputing. In particular, this section broadens the focus beyond just supercomputers, as key components of the scientific computing workflow involve storage, networking, and, notably, scientific instruments. These instruments produce large amounts of data at high rates, and analysis of this data is computationally intensive. Large-scale, highly tuned computing and workflows are required due to the need for near real-time responsiveness. Other workflows may be "wide area" and require access to outside data and databases, as in the case of a significant amount of DOE materials research. How can we develop the methods and mechanisms to validate such large and heterogeneous processes?

Section 4 focuses on *Extreme-Scale Data, Knowledge, and Analytics for Understanding and Improving Scientific Computing Integrity and Cybersecurity*. As mentioned earlier, verification is an essential part of assurance, because, as Dijkstra already pointed out decades ago [Dij70], testing shows the presence, not the absence of errors. However, due to simplifying assumptions or practical realities of verification, verification alone is insufficient as well. This section, focusing on extreme-scale analytics, emphasizes both large-scale modeling and simulation of extreme-scale scientific computing workflows as well as the analysis of static and runtime data that can provide insight into integrity failures and/or provide evidence of sound integrity. For example, such a system might enable scientists to differentiate between software and hardware errors (specifically random failures of memory access, arithmetic operations, condition tests, etc. on future exascale architectures), which will become increasingly important to ensure a reasonable level of productivity in the future.

2 Trusted Supercomputing

Develop means to build solutions for assuring the integrity of scientific computing in to the design of supercomputers.

Compared to general-purpose and enterprise computing, DOE supercomputing, along with other high-consequence computer network infrastructure of interest to DOE, already has numerous and very specific requirements that often require significant investment in research and development of specialized components, even as supercomputers contain more and more commodity parts. However, given the ways in which the DOE supercomputing culture and ecosystem already has created a culture of customized machinery optimized to meet its needs, DOE is well placed to motivate potentially substantial investment in design for integrity, correctness, and trust. For example, DOE and the broader scientific community have historically used and developed highly customized memory/processor architectures (e.g., the Tera MTA, the ASCI machines), interconnects, operating systems (e.g., the Livermore Time Sharing System, or LTSS, the Cray Time Sharing System, or CTSS), programming languages (e.g., Fortran, Unified Parallel C), compilers, and runtimes (e.g., Berkeley Lab Checkpoint/Restart) closely tied to both the hardware that computation is run on, as well as the mathematics and the algorithms used to implement that math as computation. Hence, DOE computer security R&D can take advantage of the ability to impose design constraints that would not be feasible in all systems. DOE is already pursuing co-design for exascale-capable computer systems because the useful functioning of HPC depends on optimizing complex dynamic interactions between hardware and software, especially at extreme scale. The new features added to support these HPC architectures should be evaluated for security as part of the co-design process to enable scientific computing integrity.

They should also be part of the process to make it easier for users to avoid errors that could lead to integrity loss, for example, by using programming languages or scientific workflow systems that reduce the likelihood of error leading to integrity loss. This is a unique opportunity, since DOE is engaged at the very early level stages of the system design, to introduce computer security requirements for hardware and software early in the process, especially for exascale systems. An emphasis on a co-design methodology in the development of HPC systems provides an opportunity to include integrity and security elements in the systems design.

With our drive toward exascale HPC platforms where natural faults will be increasingly commonplace, DOE is confronting the need to maintain integrity and correctness when hardware can no longer be assumed totally reliable. Such concerns differ from the commodity computing market. R&D investments are underway to address this problem, and could be uniquely leveraged to design future computing systems with intrinsic robustness that improves integrity and computer security as well. Conversely, broader advances in computer security research can suggest new resilient HPC architectures. However, detecting accidental and natural system or application errors is very different to identifying and detecting malicious users and their intent. For example, checksums (e.g., cyclic redundancy checks) or error correcting codes can detect and repair some coding errors. However, an attacker can always craft input in a way that an un-keyed checksum succeeds and the scientific computing integrity would be easily compromised. Stronger methods (e.g., keyed hashes) exist that detect both malicious use and errors, but their performance overhead is significant. Research may be directed towards investigating high performance lightweight approaches.

Sequoia, an IBM BlueGene/Q system, was the first supercomputer with more than 1 million cores. It comprises 1,572,864 cores and 1.6 petabytes of memory.

2.1 Robust and Reliable Scientific Reproducibility

Develop precise and robust means of capturing the right data to provide concrete evidence of scientific computing integrity such that reproducibility is possible and also so that integrity can be verified when it is maintained or diagnosed when it cannot.

A critical component in ensuring scientific computing integrity is the availability of high quality data about network and system behavior and performance that can be used either for near-term operational analyses or for long-term research on system integrity. Indeed, audit trails are necessary not just for ordinary security monitoring but for ongoing post hoc assurance and scientific repeatability, thus providing evidence of scientific integrity both to others in the current and future scientific community but also the interested public. Thus, audit trails for scientific computing integrity play much the same role as scientific notebooks in laboratory environments, verifiable paper ballots in public elections, and trading data in financial environments.

However, just as in these other domains, we need an understanding of what metrics these audit trails provide with regard to integrity. For example, just as risk-limiting audits of ballots cast in public elections can validate and provide statistical confidence in the results of that election [SW12], audit trails in scientific computing need to provide evidence of whatever measures of scientific computing integrity have been degraded as well. Such metrics were not conclusively defined in the workshop discussions and remain an open question. But it is clear that estimating uncertainty that could result from accidental error as well as intentional tampering would be a significant benefit to understanding scientific computing integrity.

To better clarify what constitutes useful audit data, there needs to be a dialogue with the data consumers to clearly articulate what exactly is needed, as well as what form the data should be in. Examples of this would be data about logins, process auditing, job scheduling, network traffic flow, etc. By having this dialogue with data consumers, data centers and network operators can better accommodate those consumers with the appropriate instrumentation to collect audit data, as well as storage and other resources. An additional issue regarding this would be the "sanitization" or anonymization requirements [BCP+10,NF14] for any sort of (non-internal) data sharing: how could we achieve this within an open environment?

There is also a balancing act—scientific computing integrity is one aspect of a multi-objective optimization just like currently less obscure design constraints such as speed, energy consumption, programmability, etc. And, to that end, integrity solutions may impose costs, so may be enabled or disabled depending on whether a security concern is perceived. In such a case, automated measures to improve scientific computing integrity probably require being informed by potential threat scenarios that can capture and take into account the likelihood and potential magnitude of errors due to uncertain inputs, incomplete models, incorrect implementations, silent hardware errors, and malicious tampering.

As with many operational security log systems, audit trails currently captured and used to analyze the security of DOE systems and network devices tend to be designed for human interpretation and consumption rather than automated computer analysis. This leads to the problem that either the data extraction needs to be reworked to be immediately understandable, or mechanisms need to be created which can do high-quality interpretation of systems logs at volume and velocities which are expected in large-scale networks and systems. An additional challenge is that much of the data currently gathered is not useful for answering questions about scientific computing integrity, or even security more broadly. Indeed, most security logs collected on HPC machines today, as with their general-purpose counterparts, were designed for internal debugging purposes by their own developers [PBKM05].

Finally, current approaches to scientific computing policy management rely on metadata that is easily spoofed and does not take into account the provenance of network packets. For instance, firewall and router policies may be expressed over IP addresses and ports in network packets. This information can be spoofed by an adversary and is not bound to user identities or host processes. Operational computer security staff must perform significant validation and correlation during post-exploit forensic activities in order to analyze how an adversary moved through a network. In many cases, the lack of such metadata prevents accurate forensic analysis.

The problem of audit trails is exacerbated by the increasing use of multi-component applications and more components being used in these applications, and a greater need for communication between different applications, as well as communication with services running outside of the immediate supercomputer environment (e.g., consider a database storing highly structured data). How will we create reproducible audit trails as workloads move toward greater integration with applications and services that run outside of the immediate supercomputing environment? Additional research is necessary to determine effective ways to support the auditing capabilities in HPC workloads of this type.

Audit data gathering and generation is a core issue and should be considered a fundamental challenge for verification and validation of scientific computing integrity. Such research would lead to a degree of provenance that can completely re-create the environment (including software versions, compilers, compiler flags, hardware). This would ensure reproducible science regardless of the original algorithm used to compute a mathematical function, or the round-off errors due to compiler optimization, software library version, or hardware characteristics. Additionally, new approaches are needed to tag network packets with metadata about user identities and host processes that originated the packets. The metadata should be non-spoofable and persistent, thus enabling both real-time policy enforcement as well as robust off-line forensic analysis even months after the packet captures were created. Similarly, new approaches are needed to tag data objects with metadata about user identities, application processes, data flows, etc. Challenges include issues of tagging at scale, ensuring the integrity of the metadata, and sensitivity-related challenges associated with the sharing of metadata for security purposes.

Can we protect data against malicious tampering from a storage service using secure hashes? What about signing the communication channels for source/destination authorization? How do we introduce audit trails for the passing of data between different applications, different machines, and different research groups? How do we introduce audit trails that validate the work that is being done on HPC systems is what is intended and authorized? In situ data processing will permit metadata to support data integrity to be computed at minimal cost to support data integrity as part of large-scale computations. Technologies for automating the introduction of in situ data analysis (generation of metadata for use in authentication of data integrity) and verification of inputs and outputs from large-scale scientific applications should be developed.

Though provenance tools have been created, such as Harvard's PASS system [MRBH+09], and though provenance standards are beginning to be created, such as via W3C [PROV], these tools and standards need to be embedded within data centers in a way that is clearly cognizant of the use environments, particularly including performance issues, and be expressly examined for their applicability to the unique challenges of extreme scale scientific computing integrity. Therefore, the nuanced provenance information must be captured to enable reproduction and perhaps even replay via an executable scientific workflow such as the DOE-funded Tigres project [RPH+14].

2.2 Verification and Validation for Scientific Computing Integrity

Develop techniques and tools for verifying the correctness of scientific software under performance-optimizing transformations and when executed at massive levels of parallelism.

Scientific data integrity and computer security more broadly both rest on numerous assumptions and available specifications. Improving the integrity of HPC software applications requires clear specifications of assumptions about trusted components. Specifically, with improved verification steps, the trusted components can be replaced by verified components, thus reducing the number of assumptions. This approach defines a path for verification to define improved integrity of software generally and HPC applications specifically.

Several areas in computer science have already shown substantial success in verification, such as micro-kernel verification [KEH+09], automatic theorem proving supporting the verification of parts of the Linux kernel [HJMS03], scalable verification of protocols [KNP11], and recently work on verification polyhedral codes [SLQP14]. However, there are two major aspects to be addressed for verification of scientific computing integrity for high-performance computing systems. First, verifiers must be developed that can verify given specifications of HPC application properties. To date, there has been only limited research in this area so far. Second, developers must be able to write such specifications in a certain specification language and integrate it into the development cycle. Much as with the process of developing high-assurance software systems, the workflow of developing scientific software must incorporate a cycle of specification, verification, and testing at as many layers of the software stack and for as many components as possible [PTB12].

Interestingly, domain-specific languages may even define succinct ways of incorporating or embedding necessary specifications into the applications. However, without verification we have no guarantees at all. Differentiating between software and hardware errors (specifically random failures of memory access, arithmetic operations, condition tests, etc. on future exascale architectures) will become increasingly important to ensure an acceptable level of scientific computing throughput in the future. Using a combination of automated formal verification of software and hardware, while also leveraging runtime testing will be crucial to maintaining acceptable throughput in future exascale architectures. Verification techniques for software, compilers, and data can be used to measure and assure integrity:

Software Integrity: The integrity of the software applications implementation is an essential piece and falls under the general category of software assurance. Numerous automated analyses are practical to verify low-level properties of software independent of explicit specifications. Mechanisms for users to encode further specifications of behavior and/or semantics significantly add to the sophistication of the verification and directly relate to the integrity of the application's implementation and its corresponding relationship to physics modeled by the HPC application. Examples of low-level program properties to verify would be the absence of undefined behavior specific to the programming language. Examples of useful higher-level program properties to verify would be those that can be ensured across different HPC architectures.

HPC Compiler Integrity: In order to formally prove some property of any piece of code, the correctness of a compiler must be ensured, so that one can be sure that all the properties that are proved for the source code still hold true for the binary generated from this source code. But even if the correctness of the code generation is assumed, the correctness of HPC-specific optimizations, which are essential to achieve the desired level of performance, need to verified. Since this topic is mathematically well understood, theorem proving and procedural fully automated approaches to proving can be combined to ensure the correctness of HPC specific optimizers.

Data Integrity: But supercomputers are not the only components of scientific computing. Also key is the workflow process involving scientific instruments. These instruments produce large amounts of data at high rates, and analysis of this data is computationally intensive. For example, the PDSF system at NERSC is part of the high-energy physics, astrophysics, and nuclear science workflows closely coupled with the simulation and data analysis requirements of those specific domains. Similarly, the grazing-incidence small-angle scattering (GISAXS) computation has data streaming into NERSC computing facilities from the Advanced Light Source (ALS) and requires rapid (on the order of seconds) analysis and visualization for researchers at the accelerator. Large-scale, highly tuned computing and workflows are required to do this due to the need for near real-time responsiveness. Other workflows may be "wide area" and require access to outside data and databases, as in the case of a significant amount of DOE materials research. How can we develop the methods and mechanisms to validate such large and heterogeneous processes?

Tamper-Evident Integrity Checks: To ensure scientific computing integrity, certain different kinds of checks need to be performed on stored and computed data to ensure that no data corruption has occurred, to ensure the correctness of repository contents, signatures, etc. The code that performs such an integrity check needs to be verified. In the ideal case we can verify the correctness of the code that performs the check at compile time, but perform the data-correctness check at runtime. This turns the problem into a combination of compile time and runtime verification. Certain properties of data consistency can only be checked once the data has been computed, making runtime verification an important ingredient of ensuring scientific computing integrity.

The availability of automated systems for specific forms of analysis of large-scale software (specifically binary analysis, but also source code) will be critical to an operational mechanism to verify parts of HPC systems and address essential supply chain threats. By controlling the entire stack we can implement systems that leverage the available information in source code to ease the burden on binary analysis (e.g., resolving indirect jumps is notoriously hard in a static binary analysis setting, compiler support can help here).

HPC applications have important properties in terms of precision and accuracy. Since verification can consume a vast amount of computing resources, focusing first on scientific kernels appears to be a reasonable approach. Before rushing to attempt to verify entire

HPC applications, certain properties of scientific kernels should first be specified and verified. Once these properties have been successfully verified, based on these results the verification of entire applications can be tackled. A competitive verification of scientific benchmark kernels could lead the way for the next 10 years. Then, applying the same rigor, we can proceed to verify entire applications in 20 years.

Indeed, the ability to automate the analysis of security in large-scale software and firmware at HPC scale will enable low-level certification processes to be approached—that is, automated means for determining how to carry out designs that maintain high-integrity scientific computing. Low-level certification of key elements to demonstrate the absence of security issues will be key to the mitigation of risks and mechanisms for the insurance for software. Indeed, decomposition of applications into many, possibly redundant tasks that communicate as little as possible can enhance all three by saturating the processing cores with relatively inexpensive operations that also help cross-check the computation against corruption.

2.3 Assurance of Scientific Computing Integrity Leveraging Hardware/ Software Stack Co-Design

Develop significantly improved means for balancing the assurance of scientific computing integrity between hardware and software to best monitor and maintain integrity while also minimally impacting the throughput of scientific research.

Hardware/software co-design for trustworthy supercomputing must begin to take into account computer security requirements. In some sense this is not entirely new—virtualization and containerization technologies are a noteworthy effort to demonstrate effective co-design. For example, early VMware versions (and Virtualbox and QEMU for that matter) ran "user space" guest code directly on bare hardware and relied on a kernel module to trap sensitive operations and emulate them accordingly. This process was slow and error-prone. Today, we have proper virtualization support designed directly into the hardware of our CPUs and a variety of hypervisors to leverage that support. Furthermore, para-virtualization has always been a cross-layer design approach where the guest system is aware that it is a virtual machine and can communicate with the hypervisor in a more efficient way. Intentionally breaking the virtual machine abstraction allows for higher performance. Virtualization can also strengthen the links between HPC and more general computer systems, since designs for both can mutually inform and validate each other.

Despite these past successes in general purpose computing, the co-design process needs to more broadly and deeply encompass HPC software and hardware systems to determine how the design of next-generation technologies better support integrity. There are open questions as to how best do this in HPC hardware/software co-design, given the unique architectures and constraints, but there are also opportunities. For example, as core counts increase, some cores will likely be idle during portions of an application's execution. The processing load (pressure) on individual cores will vary over time. Could supercomputing be better protected if a certain level of supercomputing resources were dedicated to analysis on an ongoing basis? Approaches for security can exploit underutilized resources. Adaptive security mechanisms that can scale to the availability of computational resources can make use of spare cycles.

Co-design would also allow us to expose security services (such as signed executable memory pages) that require kernel support to cooperating user-level applications (gradual or opt-in approach to security services, for example). This method allows users with security-sensitive workloads to leverage these capabilities based on a trade-off between scientific computing integrity and performance. What integrity features should be usefully implemented in HPC hardware? What should be done in software? How would software interact with and benefit from such hardware features? How can computer security and integrity evaluation be incorporated into modeling and analysis approaches for HPC co-design? Again, although general-purpose computing has had success with these techniques as well, there are unique challenges and opportunities in HPC environments that require substantial additional research.

3 Trust within Open, High-End Networking and Data Centers

As important as supercomputers are to DOE science, and thus assuring trustworthy supercomputing is an important part of assuring DOE scientific computing integrity, so too are high-end networking and open data centers and the integrity, management, and access to data they provide. As science and scientific communities have become larger and more distributed, the role of high-speed networks and data centers that serve diverse, international communities has increased within DOE. Furthermore, the workflows that tightly couple scientific instruments and computing facilities using networks and data storage systems are becoming increasingly important. As a result, the consequence of some aspect of the system failing and leading to a loss of scientific computing integrity in some fashion has also increased in both magnitude and likelihood. Thus, the primary aims of these high-end computing environments are to provide open, shared environments suitable for data and computationally intensive science while providing integrity during computation, experimentation, communication, and while data is at rest. At the same time, it is also vital to minimize overhead needed to maintain scientific computing. A prime example is the LHC Tier 1 laboratories in the U.S. support a large, global grid of computing and data analysis of LHC data and hundreds of DOE-supported scientists accessing, analyzing and sharing that data. Numerous other large-scale scientific experiments, both domestic and international, share similar requirements.

Contributing to the challenge of balancing throughput and scientific computing integrity, network bandwidth continues to increase exponentially and outpaces the ability of traditional computer security tools, such as deep-packet inspecting, stateful firewalls, and network intrusion detection systems, to keep pace. In particular, the space of data-compromise vulnerabilities at the instruments and storage systems seems to have been studied much less than traditional networks and computing systems. Additionally, new networking technologies such as software-defined networking (SDN) and IPv6 bring the potential to change computing and networking in ways that present both computer security challenges and opportunities for many years to come [Mon13, DOE14].

Data centers are the interfaces by which many DOE scientists interact with the data and HPC elements needed for their research. Networks are the mechanism by which these interfaces and those to critical science instruments are accessed, and by which data is transported between data centers and computing and instrument sites. As such, data centers and networks also have responsibility for scientific computing integrity, which includes limiting data modification and manipulation to authorized individuals, protecting the integrity of data at rest or in transit from malicious, accidental, and natural faults, providing confidentiality where applicable (e.g., due to scientific embargoes), and maintaining availability and ease of use so that scientific productivity is maximized.

This tension between maintaining integrity and maximizing science productivity is non-trivial as noted in the "2014 DOE High Performance Operational Review report," which makes numerous citations regarding the need for ease-of-use of computer security. However compromise of data centers can cause those data centers to be unavailable. For example, the report states:

> Current network security and data access policies pose significant challenges to data-intensive workflows. Data needs to flow seamlessly and at high performance from remote instruments to, and among, HPC centers and back to collaborators worldwide [DOEHPO].

Additionally, scientific computing integrity challenges of this research topic include:

- providing for the integrity and confidentiality of unique scientific instruments
- protecting DOE's valuable reputation to the scientific community and the public as a provider of advanced scientific facilities
- providing assurances of integrity in addition to the integrity itself.

3.1 Increasing Size and Complexity of Scientific Workflow Infrastructure

Research must be undertaken to understand the resilience of DOE scientific computing to integrity failures in order to understand how to best construct supercomputers, networks, and data centers to support increasing computing integrity.

Overall, the scale, heterogeneity, complexity, and high stakes of DOE HPC have increasingly sharpened the need for achieving computing integrity solutions applicable to supercomputing itself. At the same time, the benefits of addressing scientific computing integrity concerns for HPC can be leveraged more broadly given the continued growth of other large-scale, high-consequence systems (e.g., smart power grids) that are closely linked to HPC in their modeling, architectural design, and simulation/emulation. In both its intrinsic design and its applications, HPC is increasingly pertinent to the broader scope of computer security for high-consequence systems. There is a greater willingness on a national scale to make decisions and reach consensus on globally significant actions that may have significant economic costs and consequences not only for the U.S., but for many other nations. As a result, the data that supports these decisions is a significantly larger target for corruption and/or compromise. In addition, multi-domain workflows are now emerging where jobs need to communicate with each other. Many applications are also emerging, such as new databases to communicate results for structured data, rather than a traditional parallel file system. These databases and other services may not be running entirely inside the supercomputer. As a result, applications need to interact with services provided by the larger data centers in which the supercomputer resides, which expands the layers that need to be protected.

Moreover, networks are becoming increasingly powerful and complex, and so-called "Internet of Things" now connect a wide variety of devices. Within the context of DOE science, these advances in complex networks enable us to connect scientific facilities to supercomputers to remote storage systems to visualization sites. This complexity leads to multiple entry points to compromise the scientific data, as it is generated, transported and stored. In addition, networks may expose the science instruments to potential compromises of their data and operations in unprecedented ways. While the data integrity can be compromised at the source before it reaches data centers by certain attacks, these world-class, expensive instruments can be operationally damaged through novel attacks, such as Stuxnet [Sym11] variants.

For DOE's Energy Sciences Network, or ESnet, network traffic doubles roughly every 18 months, reflecting nearly exponential growth over the past 25 years (See Figure 2). The ESnet backbone is now 100Gbps, and nine of the large DOE labs now have 100Gbps connections. ESnet peers with several other networks at 100Gbps, and several universities now have 100Gbps connections as well. In December 2014, ESnet extended its backbone across the Atlantic Ocean by deploying three 100Gbps and one 40Gbps connections to Europe. ESnet will deploy a 400Gbps link in the San Francisco Bay Area this year, and will be adding additional 100Gbps segments as well.

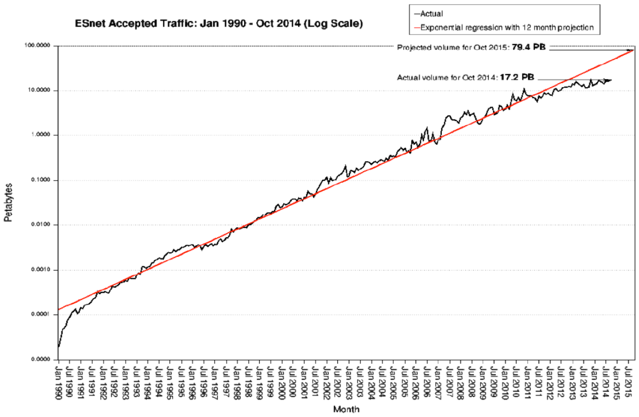


Figure 2. Exponential ESnet traffic growth over the years

Our ability to fill a 100Gbps pipe has dramatically increased as well. A single fast host with a 40Gbps NIC (network interface card) can generate a single TCP flow at 39Gbps, and a host with three of these NICS can easily fill a 100G link. Many labs and universities have started deploying 40Gbps Data Transfer Nodes (DTNs) to speed up end-to-end data transfers. 100Gbps NICs are starting to enter the market as well. The biggest vendor, Mellanox, is taking orders now and will start shipping 100Gbps NICs in Q2 2015. Security devices typically lag behind in performance by 2–3 years. While some 100Gbps firewalls exist, they tend to be very expensive, or drop packets—or both. Most firewalls are designed to handle thousands of extremely small flows, not tens of extremely large scientific flows. A 100Gbps intrusion detection system (IDS) can be built by forwarding subsets of the traffic to enough hosts, but this is also expensive and complex.

Over the past few years, the "Science DMZ" architecture has been adopted by many research institutions to take advantage of these new, faster networks [DRT+13,SDMZ]. Data-intensive science requires computers and networks that are minimally impeded by components that will slow down the scientific process, including data computation, data generation, and data storage. The Science DMZ is a framework that optimizes the network transfer aspects of data-intensive science. However this presents two key problems. First, many traditional real-time network security approaches are no longer appropriate for this architecture. For example, the devices sold commercially as firewalls often dramatically reduce network throughput due to the way that stateful firewalls doing deep packet ingestion affect TCP traffic. As a result, it is absolutely vital to research new methods that can provide appropriate security without incurring the performance impact of current firewall and other network security technologies. Second, network performance itself can contribute to loss of scientific computing integrity. Consider the large-scale scientific instruments that generate massive volumes of data and need to have data processed and down-sampled in near-real time because the original volume is too large to store on disk. In this case the real-time transmission and processing requirements of the scientific data—availability of the C-I-A triad—are vital to integrity of the scientific data because lack of availability can potentially lead to irreproducible scientific data simply being dropped from the network.

It should be noted that availability is also a risk factor. For example, as discussed earlier but worth reiterating, cybersecurity controls must be developed that do not interfere with or degrade the availability of unique-in-world capabilities, including openness and ease of access. Moreover, denial of availability on a DOE HPC facility would be unfortunate and costly. That said, a denial-of-service attack on ESnet would be disruptive to the entire $30B DOE enterprise, including both Office of Science and NNSA labs. Moreover, a compromise of ESnet could turn ESnet's network connections into "packet cannons" that could significantly adversely affect other parts of the Internet, thereby undermining ESnet's integrity.

3.2 Emerging New Network and Data Center Technologies

Research must be undertaken to explore how the evolution of virtualization, containerization, and modular runtime environments impact scientific computing integrity—where does control, layering, and modularity enhance integrity assurance and where does it add complexity and scaling problems?

A significant development in network technology in recent years is software defined networking, or SDN [DOE14]. This is an approach to computer networking that allows network administrators to manage network services through a software abstraction of lower-level functionality. This is done by decoupling the system that makes decisions about where traffic is sent (the control plane) from the underlying systems that forward traffic to the selected destination (the data plane), enabling software to compute an optimal flow routing decision on demand. SDN is commonly used at data centers in combination with network functions virtualization, or NFV, for load balancing. For example, if a virtual firewall gets overloaded, one can bring up a second instance of the firewall and send half the traffic to it instead.

From a network security perspective, the OpenFlow (OF) protocol offers researchers an unprecedented singular point of control over the network flow routing decisions across the data planes of all OF-enabled network components. Using OpenFlow, a security application can implement much more complex logic than simply halting or forwarding a flow. Such applications can incorporate stateful flow rule production logic to implement complex quarantine procedures or malicious connection migration functions that can redirect malicious network flows in ways not easily perceived by the flow participants. Flow-based security detection algorithms can also be redesigned as OpenFlow security apps, but implemented much more concisely and deployed more efficiently. While a few research efforts have explored the potential for using SDN for security applications [SPV+13, MKK11, BMP10, HTK13, ZRMB14], most of the work is preliminary, and much more work is needed to validate this approach. Separate from using SDN to enhance computer security, the actual integrity and security aspects of SDN itself are highly unexplored with several unknowns. For example, malicious users having access to controllers can be potentially damaging and recovery can be extremely difficult since device access may be cut off. Other interesting uses of SDN for integrity and computer security include findings from Mehdi et. al. [MKK11], which demonstrate one promising research direction for both endpoint shunting as well as wide area integration, and initial implementations of active security [HTK13] using SDN.

In addition to tremendous changes in the networking sphere, structural changes in the way that users, labs, and facilities interact with one another have occurred. Examples include the tremendous growth in virtualization and containerized services, huge changes in the effective use of multi-core and GPU offloading, and the emergence of exascale systems as an immediate reality.

In addition to complexity due to scale, there has been a significant increase in the complexity of the systems deployed, marked by increasing virtualization and abstraction, including such technologies as Science

DMZ, Docker, Linux Containers, and SDN. Together these yield a landscape typified by more complex workflows over distributed high-capacity networks with deep layering of indirection and authentication. While increasing efficiency, capacity, and flexibility, and in some ways improving security, these also introduce the potential for new vulnerabilities, either individually or collectively due to composition and complexity.

For example, the use of Docker [Dock] containers, which allow a systems administrator to easily create a secure sandbox for a particular network service, have been widely adopted. While this makes the service more secure by reducing the attack service, the additional abstraction layers can make it even harder to do forensics after an attack. More work is needed on how to model attacks in a containerized world. Thus containerization strategies bring both new challenges and opportunities with regard to scientific computing integrity.

Site architectural changes have become more significant as well, with the continuing development of virtual entities such as the DOE Systems Biology Knowledgebase [kbase] which sit (from a computational perspective) somewhere between totally outside and inside the traditional perimeter. Discussions regarding inter-lab and facility collaboration are beginning to push the traditional boundaries with data storage, file system, and job submission. All of these things have begun to wear away at the traditional notion of a network border where all activity happens within a well-defined address space. User interaction with facilities have also undergone tremendous change, moving from a command line-only schema to web services such as the NERSC Web Toolkit (NEWT)[NEWT] which provides browser access to System Status, File Upload/Download, Directory Listings, Command Execution, Batch Queue Jobs, Accounting Information and Persistent object storage. The nature and scale of collaboration is also changing, as exemplified by the near real-time analysis from the experiment at the Linac Coherent Light Source (LCLS) at SLAC using NERSC resources to make more effective use of valuable LCLS beam time, thereby enhancing the scientific value of the facilities' resources [ESWork].

We advocate the creation of a scalable system that enables and demonstrates concurrent, dynamic, and time-constrained multilevel security for exascale computing that minimizes impact to performance and usability while meeting information protection requirements.

3.3 Identity and Access Management to Support Scientific Collaboration

Means for developing coherent means for authorization and access controls particular to the open science mission can maximize integrity and computing efficiency

Identity and access management is critical to expressing the organization of scientific collaborations regarding who can access data, who can control instruments, who has priority to resources, etc. It is also faces a number of challenges, including:

- Passwords continue to be a primary authentication mechanism but are increasingly susceptible to attack, leading to the impersonation of legitimate users as one of, if not the, most common computer security failures seen. Two-factor or multi-factor authentication and biometrics are becoming more common, but are not a silver bullet to this problem [BHvOS12].
- As scientists, similar to most users of the Internet and World Wide Web, obtain more and more accounts, the number of passwords increasingly stretches the ability of human memory, leading to organizational deployment of federated identity (e.g., InCommon), and users' personal use of social identities (e.g., Google, Facebook) and password managers.
- Users are increasingly using multiple devices (desktops, laptops, smart phones, tablets, etc.) with each authenticating as the user.
- As DOE scientific collaborations have grown, roles and the privileges granted those roles within those communities have become more complex.
- Scientific collaborations are playing a large and increasing role fulfilling key aspects of identity and access management [CJWC14].

To support scientific integrity, identity and access management also needs to keep pace with the evolution of computer security generally, which has evolved over the past two decades from being based on prevention to being increasingly based on managing risks to the business mission of organizations. Today, the process of managing risks is done relatively statically at the programmatic level. The organization evaluates its computational assets based on their importance to its mission, the threats based on a best understanding of the landscape, and then allocates computer security assets statically to provide the best protection, detection, and response capabilities to manage risk at an acceptable level.

A more complete benefit of a risk-based approach can be achieved by implementing an understanding of risk and the ability to react to increased risk into the operational context. For example, risk-based authentication [JR04, PTK13] ties the required strength of authentication to the level of risk of the requested access. This has been implemented in at least one supercomputing facility by examining the geolocation of the client and the history of where they have connected from in the past. To continue to adapt to the rapidly changing technology landscape as well as our threats, a future computer security architecture will need to take into account an understanding of organizational risk in the operational context and allow for dynamic application of computer security controls.

4 Extreme-Scale Data, Knowledge, and Analytics for Understanding and Improving Scientific Computing Integrity and Cybersecurity

The size and importance of future DOE HPC systems require that computational resilience be designed and built into those systems. By this we mean building in the ability for systems to quickly and automatically recover from component failures and malicious attacks, specifically protecting the integrity of the computations and data resident on DOE HPC systems.

Large-scale heterogeneous systems, such as exascale-generation HPC, distributed smart grid deployments, and current and future DOE high-end scientific user facilities, clearly involve increasing degrees of complexity, introducing significantly greater opportunity for component failures, as well as compounding statistical error rates between components. This increase in error and uncertainty complicates modeling and analysis. In particular, this additional noise provides "cover" for malicious activity over computer networks, making classification of errors as operational failures or deliberate malicious activity challenging for traditional approaches. New approaches to provide accurate and useful HPC system monitoring, analysis and recovery are needed.

Specifically, advances in automated model generation, causal inference, and metrics for HPC security are all necessary to provide useful decision-making in exascale security. This decision-making will need to be informed by new approaches to the analysis, characterization, and risk assessment of possible attacks against HPC systems.

The current state of the art, future possibilities, and potential impact of investment in these areas of modeling and simulation of HPC systems for scientific computing resilience are presented in the following sections. Of particular interest is the use of large-scale analytics for detecting integrity loss in current and future HPC systems.

4.1 Modeling HPC Systems and their Operating Environments

Develop a framework for collecting data from multiple sources at an unprecedented scale that collectively represent the system under study to enable adaptive, streaming analysis for monitoring and maintaining large-scale scientific computing integrity.

Building a model of a complex engineered system is a standard method to predict, verify and test the features of the system in many different domains such as nuclear reactors, aircraft, and communication networks,. To date, however, a system-level science [FK06] approach has not been applied to computing infrastructures and systems, their behaviors, or their vulnerabilities. Comprehensive, holistic, system-wide models are necessary to predict, verify, and test the security features of a complex engineered system, such as an HPC cluster or the power grid. Such a model should ideally quantify in real-time the general security state of the system.

Taking the example of an HPC cluster, we envision models of supercomputers that can be used to: **1** predict runtime performance of an application code portfolio; **2** detect potentially malicious deviations from detailed specifications within hardware, middleware, and software in early deployment phases of a new platform; and; **3** help flag anomalous user or code behavior during regular use in the lifecycle of a supercomputer platform. With the transition to exascale computing, the scale of the system will require an integrated, automatic analytic environment that provides a whole-system view. Such a whole-system model approach requires integrated models of all relevant subcomponents as well as model elements that integrate the subcomponent models. In the case of supercomputers, this would include models of the compute nodes, the interconnection communication network, the software stacks including application software, the filesystem, the access nodes, and the behavior of the users (both normal and malicious), and even the physical security, HVAC and power systems that are part of the cluster environment.

One particular area worth further investigation is the development of large-system models with surrogate components. Exascale HPC, smart grid technologies, and other large-scale heterogeneous systems all include components that interact in complex ways at multiple levels of granularity (e.g., processor, node, blade, system). Additionally, these systems are composed of components that are qualitatively different from each other, e.g., physical, computer-based, and human. Surrogate components are estimators used to model the behavior of such components, which are then combined to model the behavior of the entire system.

This approach has been applied successfully to real-world problems [ZCZ+13]. However, progress is needed to expand this approach to heterogeneous/hierarchical systems at multiple levels of granularity. Expanding on work in adaptive models [WDY+14] could provide a "plug and play" modeling capability, allowing the modeling framework to adapt in real time to changes to the actual system. Finally, we need to fully utilize the background knowledge available about the science behind these large-scale systems, ensuring that, where possible, physical laws and established principles are incorporated to improve estimator accuracy and inform the combination of model components.

Advances in computing and the availability of big data should allow us to efficiently build models and algorithms applied to big, dynamic, noisy, uncertain domains such as the Internet. Preliminary hints at the potential of this approach are IBM's Watson machine [Cho01] and extreme-scale discrete event simulation efforts [NBB+13, SYF10, MJV+14]. The most detailed of such holistic system models will present a scalability challenge by themselves, thus requiring significant supercomputing resources for execution. This calls for a fortunate symbiosis in the modeling relationship: in addition to modeling complex systems such as supercomputers, we use will use supercomputers to model other complex systems including themselves.

It should be noted that aspects of this have been done previously to detect misuse on supercomputers. As noted earlier, regularity of behavior patterns on supercomputers versus conventional computing platforms is unique to DOE. This has previously led to success in past efforts to "fingerprint" what is running on supercomputers and verify that it is within policy for what a user is supposed to be running on DOE resources [Pei10, SP10, WEPB12, WPB13], for example whether a process is "mining bitcoins" or performing some other cryptographic operations, or processing data that is not permitted on an open, unclassified scientific computing platform.

A comprehensive approach to decision-making on the Internet requires maintaining knowledge of heterogeneous components at multiple levels of granularity over multiple time scales, and performing integrated analysis on the data produced from the interactions between all of them. When looking at systems at scale, this huge production of raw data requires unprecedented computational resources to ingest, analyze, incorporate into models, draw inferences from, and make decisions on. Often these decisions, in the context of computer security events, are extremely time sensitive. We envision a data collection and analysis framework that is capable of collecting data at an unprecedented scale from multiple sources that collectively represent the system under study. Such a framework should be extensible, adaptive, and streaming, updating its knowledge base of facts through generation and scoring of multiple hypotheses, "making decisions" by synthesizing across the various hypotheses, and acting by deploying the best approach determined by informed risk assessments.

4.2 Automated Learning of and Reasoning on HPC Models

Develop means to learn and maintain interdependent causal models of the scientific computation, exascale system, and computer security in real-time to enable better, faster recovery to reduce disruptions to scientists' efforts.

As discussed above, aggregated, abstracted models of exascale scientific computations will be extremely important for computational and resiliency performance monitoring and correction. Because of continual software development and tuning, varying input dataset characteristics and runtime computation-to-processor mapping variations, the computational and semantic modeling of exascale computations might most effectively be achievable through the automated machine learning of granular models.

Recent progress in machine learning applied to patterns (such as deep learning)[HOT06], program invariants [GLMN13, ZMAA13], compressive sensing of spatio-temporal data (signal processing)[Bar07], network science (characterizing graph and network structures)[HKB+12] and behavior learning (scalable learning of automata from observations)[CC11] should be leveraged to automate such exascale program and system modeling. By the same token, many of these techniques are very compute-intensive and presently constrained by computational resources so that exascale computing will be needed itself to scale such techniques to apply to exascale computing.

Advancement in such real-time algorithms for automatically learning appropriate computational workflows and attacks against them from exascale data should be combined with risk models of attacks against those workflows. This would provide a capability for scientists to trust their computing infrastructure, data and, ultimately, the results of their most critical simulations.

HPC system models need to be augmented with causation, reasoning, and explanation capabilities. Only then can we reason about the security of a complex, networked computer system at its various levels (social, human, roles, information, network, and the real world [BSW14]) and consequently manage the integrity of the scientific computation and data. Indeed, the lack of reasoning in previous analytics for computer security has made the computer security domain one of the few computing domains that still heavily relies on human reasoning and explanation.

Advances in computing and the availability of big data have only recently allowed us to research causation on the Internet. An example of previous work on causation for computer security can be found in Mugan [Mug13], where he uses a dynamic Bayesian network to learn an attack tree. Another example is Xie et al.'s work [XLO+10] where they capture the uncertainty inherent in computer security of enterprise domains by using Bayesian networks. Other approaches for using Bayesian methods to leverage distributed security data have been constrained by the need for significantly more computing and networking resources to support the analysis [BWC02]. Such constraints may not be present in current and future HPC systems. As a result, running large-scale, accurate causal models and reasoning algorithms in real-time to produce system security state estimates and associated explanations is now becoming possible and lies squarely within the purview of ASCR.

Additionally, classification of scientific integrity failures is an important capability because correct classification enables identification of the appropriate recovery strategies. Since not all faults cause the same damage, fault classification methods are often accompanied by the risks associated with each fault. Ye et al.[YNF06] provide a classification taxonomy for computer attacks and associate risks. Current classification methods and recovery strategies do not take into account the dynamic aspects of the changing supercomputing environment involving human users and possible physical instruments inputting data in real-time. These novel challenges are exacerbated by the complexity of the science being modeled and the complexity of exascale systems.

The identification of scientific integrity failures and recovery strategies should be placed close to the sensors on an exascale system. Specifically, the classification and recovery strategies should be integrated within the exascale system. In this way, the faults can be classified faster because the analytics will be closer to the sensors (alleviating latency issues). Moreover, recovery strategies can be automatically triggered close to the fault and not affect other parts of the system. Research is needed to develop better, faster classification that will lead to better, faster recovery and reduce the disruptions of scientists' efforts. Such research will involve efforts in both exascale hardware, placement of sensors and analytics (for minimizing latency), and modularized recovery strategies.

Some research questions under this area are:

- What are the appropriate semantics (i.e., language) for explanation in the computer security domain? Examples of explanations include reasoning about similar behavior in various parts of the computer system. Does each layer of computers and networks need its own specific language?
- Can the existing causal models and reasoning algorithms accurately and efficiently capture the volatile, adversarial, and interdependent nature of internetworked systems? If not, what should the new causal models look like?
- Are big causal models needed for computer security? Or can many small causal models be effective?
- Are there projections into an interpretable lower-dimensional space, where causal models can be learned more easily from big data?
- How should the causal models of scientific computation and their data interact with the casual models of the exascale system and its security?

4.3 Metrics for HPC integrity

Develop metrics to model, quantify, and manage exascale performance to allow exascale computing users and system operators to effectively manage the tradeoffs between scientific throughput and scientific computing integrity performance.

Exascale computing performance has multiple dimensions, including classical computing performance metrics (operations executed per second and data transfer rates, for example) and integrity levels of the actual scientific computation and data. Effective exascale computing ideally seeks high performance in all dimensions. But this may not be possible due to tradeoffs between observed computational performance and scientific computing integrity arising from the overhead required by extraneous code that enforces, monitors, and analyzes integrity. Defining, modeling, and measuring these performance dimensions is a major challenge for high-confidence exascale computing systems. Moreover, understanding and managing tradeoffs between such performance dimensions is necessary for operating effective exascale systems.

Quantifying, modeling and measuring classical computing performance are relatively mature areas that can be leveraged immediately [HP12]. The same cannot be said about the integrity and reliability properties of general computing systems, let alone large-scale open scientific computations [Jan11] such as envisioned for future exascale systems. While several memory and data transfer error detection and correction mechanisms already exist, those mechanisms assume small, independent failures appropriate in reliability analysis and recovery for natural, organic faults in the computing system [CT12].

However, such mechanisms are not appropriate for either detecting or recovering from large, correlated errors that can be introduced deliberately by human adversaries with the intent of subverting the integrity and/or efficiency of an important scientific computation. Quantifying and measuring the operational resiliency of an open, heterogeneous exascale computing system is an unexplored research area. Significant progress on the development of such capabilities would allow exascale computing users and system operators to effectively manage tradeoffs between integrity and overall computational performance.

4.4 Risk Assessment and Management of HPC Integrity

Research is required to develop new methods for meaningful risk measures and threat measures of HPC integrity.

Quantifying scientific computing integrity will allow the management of processes on any DOE scientific instrument or facility. HPC behavioral models, together with methods to dynamically update these models in real-time as behaviors change, will feed into algorithms that generate predictive threat and risk assessments that will in turn feed into mitigation and recovery actions. User behavior models could include temporal (when or with what kinds of delays does the user perform certain actions?), probabilistic graphical models capturing the user's normal behavior and may also include geospatial information (where does the user perform those actions?).

Threat models could include temporal probabilistic graph models with spatial attributes (such as stochastic temporal automata)[PSTM14], timed Petri nets [PC06] and Petri net models [CBK09]). Exascale algorithms that leverage HPC resources to automatically learn these types of models in real-time will be needed—and they will need to operate not just on historical data, but live real-time streaming data. Algorithms to incrementally update these models in real-time are also critical. Some of the data that comes in may be inherently noisy and uncertain.

An additional approach is to leverage the notion of self-protecting data [DOE08], that is data objects capable of protecting themselves from various kinds of threats. However, most of the work done so far on self-protecting data has focused on protecting the confidentiality [CJL12, SGS+00], whereas the current DOE focus is on integrity. An even more promising direction is integrating the notion of self-protecting data with concepts such as chain of custody and digital rights management, and with anti-tamper technologies, as discussed earlier with regard to provenance and audit trails. Ultimately, we would like to think about data as "smart data objects" which, beyond merely storing some piece of information, have the capability of duplicating themselves, executing code on the data, detecting and responding to attempts to maliciously alter the data.

5 Epilogue

This report has provided an analysis of the DOE's needs for new applied, computational, and mathematical developments in order to support the science- and engineering-based solutions to the problems of computer security and scientific computing integrity that are of critical national importance now and in the future. Given the DOE's energy, environmental, and national security missions, the DOE Advanced Scientific Computing Research (ASCR) Division has a vital need to assure scientific computing integrity in order to help assure the results of the scientific research itself. The importance is even greater when the scientific research can affect national policy decisions and commercial development, as DOE Office of Science research often can. At the same time, given the impending transition to exascale computing, there is also a significant opportunity for ASCR to build security and integrity assurance into exascale systems (and beyond) by starting now to research the means for doing so. By starting this research and development now, ASCR will provide the basis for assuring extreme-scale scientific computing integrity as it moves well into the 21st century, continuing its heritage and legacy of large-scale scientific integrity, while also developing techniques that will undoubtedly have application beyond the DOE Office of Science as well.

References

[AMP+14] M. Albanese, C. Molinaro, F. Persia, A. Picariello, and V.S. Subrahmanian. Discovering the Top-k Unexplained Sequences in Time-Stamped Observation Data." *IEEE Transactions on Knowledge and Data Engineering*, 26.3: 577–594, 2014.

[Ami02] M. Amin. Security challenges for the electricity infrastructure. *Computer*, 35(4):8–10, 2002.

[Bar07] R. G. Baraniuk. Compressive sensing. *IEEE Signal Processing magazine*, 24.4, 2007.

[BB09] J. M. Brase and D. L. Brown. Modeling, Simulation and Analysis of Complex Networked Systems: A Program Plan, May 2009.

[BCP+10] M. Bishop, J. Cummins, S. Peisert, A. Singh, D. Agarwal, D. Frincke, and M. Hogarth. Relationships in Data Sanitization: A Study in Scarlet. In *Proceedings of the 2010 New Security Paradigms Workshop*, pages 151–164, Concord, MA, September 21–23, 2010.

[BEF+10] M. Bishop, S. Engle, D. A. Frincke, C. Gates, F. L. Greitzer, S. Peisert, and S. Whalen. A Risk Management Approach to the 'Insider Threat'. In Christian W. Probst, Jeffrey Hunker, and Matt Bishop, editors, *Insider Threats in Cybersecurity, Advances in Information Security Series*, pages 115–138. Springer, Berlin, September 2010.

[BHvOS12] J. Bonneau, C. Herley, P. C. van Oorschot, and F. Stajano. The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes. In *Proceedings of the 33rd IEEE Symposium on Security and Privacy*, pages 553–567, May 2012.

[Bib77] K. Biba. Integrity Considerations for Secure Computer Systems. Technical Report MTR-3153, MITRE Corporation, Bedford, MA, April 1977.

[BL73] D. E. Bell and L. J. LaPadula. Secure Computer System: A Mathematical Model. Technical Report 2547, Volume II, MITRE, 1973.

[BH09] P. Burnap and J. Hilton. Self Protecting Data for De-perimeterised Information Sharing. *Proc. of the Third International Conference on Digital Society (ICDS'09)*, IEEE, 2009.

[BMP10] R. Braga, E. Mota, and A. Passito. Lightweight DDoS flooding attack detection using NOX/OpenFlow. *35th IEEE Conference on Local Computer Networks (LCN)*. IEEE, 2010.

[BSW14] A. Barnett, S. R. Smith, and R. P. Whittington. Using causal models to manage the cyber threat to C2 agility: Working with the benefit of hindsight. In *Proc. of the 19th ICCRTS*, Paper 081, 2014.

[BWC02] D. J. Burroughs, L. F. Wilson, and G. Cybenko. Analysis of distributed intrusion detection systems using Bayesian methods. *21st IEEE Conference on International Performance, Computing, and Communications*, IEEE, 2002.

[CAL+11] A. A .Cárdenas, S. Amin, Z. Lin, Y.Huang, C.Huang, and S. Sastry. Attacks against process control systems: risk assessment, detection, and response. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, pages 355–366. ACM, 2011.

[CBK09] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)* 41.3: 15, 2009.

[CC11] G. Cybenko and V. Crespi. Learning hidden markov models using nonnegative matrix factorization. *IEEE Transactions on Information Theory*, 57.6, pages 3963–3970, 2011.

[CJL12] Y. Chen, P. A. Jamkhedkar, and R. B. Lee. A software-hardware architecture for self-protecting data. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*. ACM, 2012.

[Cho01] T.-S. Chou. Cybersecurity threats detection using ensemble architecture. *International Journal of Security and Its Applications*, 5(2), 2001.

[CJWC14] R. Cowles, C. Jackson, V. Welch, and S. Cholia. A Model for Identity Management in Future Scientific Collaboratories. *International Symposium on Grids and Clouds (ISGC)*, 2014.

[CT12] T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[Den76] D. E. Denning. A lattice model of secure information flow. *Communications of the ACM*, 19(5):236–243, May 1976.

[Dij70] E. W. Dijkstra. *Notes on Structured Programming*, Technological University Eindhoven, Netherlands, 1970.

[Dock] Docker. https://www.docker.com

[DOE08] C.Catlett, et al., editors. A Scientific Research and Development Approach to Cyber Security Submitted to the Department of Energy On behalf of the Research and Development Community, November 2008.

[DOE14] Software Defined Networking for Extreme-Scale Science: Data, Compute, and Instrument Facilities, Report of ASCR Intelligent Network Infrastructure Report, http://www.orau.gov/ioninfrastructure2014/, January 2015.

[DOEHPO] DOE High Performance Operational Review. 2014: http://www.osti.gov/scitech/biblio/1163236

[DRT+13] E. Dart, L. Rotman, B. Tierney, M. Hester, and J. Zurawski. The Science DMZ: A Network Design Pattern for Data-Intensive Science. *In Proceedings of the IEEE/ACM Annual SuperComputing Conference (SC13)*, Denver CO, 2013.

[EIA11] Energy Information Administration. Electric systems respond quickly to the sudden loss of supply or demand. http://www.eia.gov/todayinenergy/detail.cfm?id=3990, 2011.

[ESWork] ESnet, Multi-facility Workflow Case Study, https://www.es.net/science-engagement/case-studies/multi-facility-workflow-case-study/

[FB04] D. Frincke and M. Bishop. Guarding the castle keep: Teaching with the fortress metaphor. *IEEE Security & Privacy*, 2(3): 69–72, 2004.

[FK06] I. Foster and C. Kesselman. Scaling system-level science: Scientific exploration and IT implications. *IEEE Computer*, 39(11): 31–39, 2006.

[GCH+13] P. Garg, C. Gentry, Halevy, Raykova, Sahai, and Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *Proceedings of the 54th IEEE Annual Symposium on Foundations of Computer Science*, 2013

[Gen09] C. Gentry. Fully homomorphic encryption using ideal lattices. *STOC*, Vol. 9, 2009.

[GLMN13] P. Garg, C. Löding, P. Madhusudan, and D. Neider. Learning universally quantified invariants of linear data structures. *Computer Aided Verification*. Springer Berlin Heidelberg, 2013.

[GM82] J. A. Goguen and J. Meseguer. Security Policies and Security Models. In *Proceedings of the 1982 Symposium on Security and Privacy*, pages 11–20, April 26–28, 1982.

[HKB+12] D. Havlin, D.Y. Kenett, E. Ben-Jacob, A. Bunde, R. Cohen, H. Hermann, J.W. Kantelhardt, J. Kertész, S. Kirkpatrick. J. Kurths, J. Portugali, and S. Solomon. Challenges in network science: Applications to infrastructures, climate, social systems and economics. *European Physical Journal-Special Topics*, 214.1, pages 273, 2012.

[HOT06] G. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18.7, pages 1527–1554, 2006.

[HJMS03] T. A. Henzinger, R. Jhala, R. Majumdar, and G.Sutre. Software verification with blast. *In Model Checking Software*, pages 235–239. Springer, 2003.

[HP12] J. L. Hennessy and D. A. Patterson. *Computer architecture: a quantitative approach*. Elsevier, 2012.

[HRU76] M. A. Harrison, W. L. Ruzzo, and J. D. Ullman. Protection in Operating Systems. *Communications of the ACM*, 19(8):461–471, August 1976.

[HTK13] R. Hand, M. Ton, and E. Keller. Active security. In *Proceedings of the Twelfth Workshop on Hot Topics in Networks*. ACM, 2013.

[Jan11] W. Jansen. Research Directions in Security Metrics. *Journal of Information System Security*, 7.1., 2011.

[JR04] A. K. Jain and A. Ross. Multibiometric systems. *Communications of the ACM*, 47(1): 34–40, 2004.

[kbase] KBase: The Department of Energy Systems Biology Knowledgebase, http://kbase.us

[KEH+09] G. Klein, K. Elphinstone, G. Heiser, J. Andronick, D. Cock, P. Derrin, D. Elkaduwe, K. Engelhardt, R. Kolanski, M. Norrish, T. Sewell, H. Tuch, and S. Winwood. seL4: Formal Verification of an OS Kernel. In *Proceedings of the 22nd ACM Symposium on Operating Systems Principles*, pages 207–220, 2009.

[KNP11] M. Kwiatkowska, G. Norman, and D. Parker. Prism 4.0: Verification of probabilistic real-time systems. *In Computer-aided Verification*, pages 585–591. Springer, 2011.

[LPS13] P. Lions, B. Perthame and P. E. Souganidis. Scalar conservation laws with rough (stochastic) fluxes. *Stochastic partial differential equations: analysis and computations*, 1.4, pages 664–686, 2013.

[MKK11] S. A. Mehdi, J. Khalid, and S. A. Khayam. Revisiting traffic anomaly detection using software defined networking. *Recent Advances in Intrusion Detection*. Springer Berlin Heidelberg, 2011.

[MJV+14] S. M. Mniszewski, C. Junghans, A. F. Voter, D. Perez, S. J. Eidenbenz. ADSim: Discrete Event-based Performance Prediction for Temperature Accelerated Dynamics. *ACM TOMACS Journal* Issue 25:3, 2014.

[MMCS11] A Multifaceted Mathematical Approach for Complex Systems – Report of the DOE Workshop on Mathematics for the Analysis, Simulation, and Optimization of Complex Systems. September 13–14, 2011.

[Mon13] I. Monga, et al. Operationalization of Software-Defined Networks (SDN) Program Review, Dec. 16–17, 2013.

[MPS14] C. McParland, S. Peisert, and A. Scaglione. Monitoring Security of Networked Control Systems: It's the Physics. *IEEE Security & Privacy*, 12(6), Nov/Dec 2014.

[MRBH+09] K. Muniswamy-Reddy, U. Braun, D. A. Holland, P. Macko, D. Maclean, D.l Margo, M. Seltzer, and R. Smogor. Layering in Provenance Systems. In *Proceedings of the 2009 USENIX Annual Technical Conference*, pages 10–23, 2009.

[Mug13] J. Mugan: A developmental approach to learning causal models for cybersecurity. *SPIE Defense, Security, and Sensing*, 2013.

[NBB+13] S. Nikolaev, P. D. Barnes, J. M. Brase, T. W. Canales, D. R. Jefferson, S. Smith, R. A. Soltz, and P. J. Scheibel. Performance of Distributed ns-3 Network Simulator. In *Proceedings of SIMUTools 2013, ICST/ACM*, 2013.

[NEWT] NERSC Web Toolkit (NEWT) https://newt.nersc.gov/

[NF14] Arvind Narayanan and Edward W. Felten. No silver bullet: De-identification still doesn't work, July 9, 2014.

[OFC14] J. Oliver, S. Forman and C. Cheng. Using Randomization to Attack Similarity Digests. *Applications and Techniques in Information Security*. Springer Berlin Heidelberg, pages 199–210, 2014.

[PBKM05] S. Peisert, M. Bishop, S. Karin, and K. Marzullo. Principles-Driven Forensic Analysis. In *Proceedings of the 2005 New Security Paradigms Workshop (NSPW)*, pages 85–93, Lake Arrowhead, CA, October 2005.

[PC06] V. Parameswaran and R. Chellappa. View invariance for human action recognition. *International Journal of Computer Vision*, 66.1: 83–101. 2006

[Pei10] S. Peisert. Fingerprinting Communication and Computation on HPC Machines. *Technical Report LBNL-3483E*, Lawrence Berkeley National Laboratory, June 2010.

[PH11] D. Poole and W. Hereman. Symbolic computation of conservation laws for nonlinear partial differential equations in multiple space dimensions. *Journal of Symbolic Computation*, 46.12, pages 1355–1377, 2011.

[PROV] PROV Model Primer, W3C Working Group Note 30, April 2013. http://www.w3.org/TR/prov-primer/

[PSTM14] A. Pugliese, V.S. Subrahmanian, C. Thomas and C. Molinaro, PASS: A Parallel Activity Search System, *IEEE Transactions on Knowledge & Data Engineering*, 26(8): 1989–2001. 2014.

[PTB12] S. Peisert, E. Talbot, and M. Bishop. Turtles All The Way Down: A Clean-Slate, Ground-Up, First-Principles Approach to Secure Systems. In *Proceedings of the 2012 New Security Paradigms Workshop (NSPW)*, pages 15–26, Bertinoro, Italy, September 19–21, 2012.

[PTK13] S. Peisert, E. Talbot, and T. Kroeger. Principles of Authentication. In *Proceedings of the 2013 New Security Paradigms Workshop (NSPW)*, pages 47–56, Banff, Canada, Sept. 9–12 2013.

[PZM12] R. Poliker, C. Zhang, and Y. Ma: *Ensemble Machine Learning: Methods and Applications*. Springer, 2012.

[RPH+14] L. Ramakrishnan, S. Poon, V. Hendrix, D. Gunter, G. Pastorello, and D. Agarwal. Experiences with user-centered design for the Tigres workflow API. *Proceedings of the 10th IEEE International Conference on e-Science (e-Science)*, 2014.

[SDMZ] Science DMZ – A Scalable Network Design Model for Optimizing Science Data Transfers https://fasterdata.es.net/science-dmz/

[SGS+00] J. D. Strunk, G. Goodson, M. Scheinholtz, C. Soules, and G. Ganger. Self-securing storage: protecting data in compromised system. In *Proceedings of the 4th conference on Symposium on Operating System Design & Implementation-Volume 4*. USENIX Association, 2000.

[SHC05] K. Sun, Z. Han, and Y. Cao. Review on Models of Cascading Failure in Complex Power Grid [J]." *Power System Technology* 13: 1–9, 2005.

[SPV+13] S. Shin, P. Porras, V. Yegneswaran, M. Fong, G. Gu, and M. Tyson. FRESCO: Modular Composable Security Services for Software-Defined Networks. NDSS. 2013.

[SLQP14] M. Schordan, P.Lin, D. Quinlan, and L.Pouchet. Verification of polyhedral optimizations with constant loop bounds in finite state space computations. In *Leveraging Applications of Formal Methods, Verification and Validation. Specialized Techniques and Applications*, pages 493–508. Springer, 2014.

[SP10] R. Sommer and V. Paxson. Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. In *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, CA, May 2010.

[SW12] P. B. Stark and D. Wagner. Evidence-based elections. *Security & Privacy*, IEEE, 10(5):33–41, 2012.

[SYF10] N. Santhi, G. Yan, and S. Eidenbenz. CyberSim: Geographic, temporal, and organizational dynamics of malware propagation. Simulation Conference (WSC), 2876–2887. 2010.

[Sym11] Symantec, W32.Stuxnet Dossier (v.1.4), Feb. 2011. http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf

[WDY+14] C. Wang, Q. Duan, W., A. Ye, Z. Di, C. Miao. An evaluation of adaptive surrogate modeling based optimization with two benchmark problems, *Environmental Modelling & Software*, Volume 60, pages 167–179, October 2014.

[WEPB12] S. Whalen, S. Engle, S. Peisert, and M. Bishop. Network-Theoretic Classification of Parallel Computation Patterns. *International Journal of High Performance Computing Applications (IJHPCA)*, 26(2): 159–169, May 2012.

[WPB13] S. Whalen, S. Peisert, and M. Bishop. Multiclass Classification of Distributed Memory Parallel Computations. *Pattern Recognition Letters (PRL)*, 34(3): pages 322–329, February 2013.

[XLO+10] P. Xie, J. H. Li, X. Ou, P. Liu, and R. Levy. Using Bayesian networks for cybersecurity analysis. In *Proc. of the 2010 IEEE/IFIP Int'l Conf. on Dependable Systems and Networks (DSN)*, pp. 211–220, 2010.

[YNF06] N. Ye, C. Newman, and T. Farley. A system-fault-risk framework for cyber attack classification. *Information Knowledge Systems Management*, vol. 5, pp. 135–151, 2006.

[ZCZ+13] J. Zhang, S. Chowdhury, J. Zhang, A. Messac and L. Castillo. Adaptive Hybrid Surrogate Modeling for Complex Systems. *AIAA Journal*, Volume 51, 3, page 643, 2013.

[ZMAA13] Y. Zhang, D.S. Myers, A.C. Arpaci-Dusseau, and R.H. Arpaci-Dusseau. Zettabyte reliability with flexible end-to-end data integrity. In *Proceedings of IEEE 29th Symposium on Mass Storage Systems and Technologies (MSST)*. IEEE, 2013.

[ZRMB14] A. Zaalouk, R. Rhondoker, R. Marx, and K. Bayarou. OrchSec: An orchestrator-based architecture for enhancing network-security using Network Monitoring and SDN Control functions. *2014 IEEE Network Operations and Management Symposium (NOMS)*, IEEE, 2014.

[ZWZ12] Z. Zhang, J. Wang and H. Zha. Adaptive manifold learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34.2, pages 253–265, 2012.

Appendix: Workshop Organizing Committee

Workshop Co-Chairs:

Sean Peisert	Lawrence Berkeley National Laboratory and University of California, Davis
George Cybenko	Dartmouth College
Sushil Jajodia	George Mason University

Topic Chairs:

David L. Brown	Lawrence Berkeley National Laboratory
Christopher DeMarco	University of Wisconsin-Madison
Paul Hovland	Argonne National Laboratory
Sven Leyffer	Argonne National Laboratory
Celeste Matarazzo	Lawrence Livermore National Laboratory
Stacy Prowell	Oak Ridge National Laboratory
Brian Tierney	Energy Sciences Network (ESnet)
Von Welch	Indiana University

Appendix: Workshop Participants and Other Contributors

Massimiliano Albanese	George Mason University
Jon Bashor	Lawrence Berkeley National Laboratory
Michael Berry	University of Tennessee, Knoxville
David L. Brown	Lawrence Berkeley National Laboratory
Scott Campbell	National Energy Research Scientific Computing Center
Stephen Crago	University of Southern California
George Cybenko	Dartmouth College
Jon DeLapp	Integrity Consulting Solutions
Christopher DeMarco	University of Wisconsin-Madison
Jeff Draper	University of Southern California
Manuel Egele	Boston University
Stephan Eidenbenz	Los Alamos National Laboratory
Tina Eliassi-Rad	Rutgers University
Ryan Goodfellow	University of Southern California/ISI
Paul Hovland	Argonne National Laboratory
Sushil Jajodia	George Mason University
Cliff Joslyn	Pacific Northwest National Laboratory
Alex Kent	Los Alamos National Laboratory
Sven Leyffer	Argonne National Laboratory
Robert Lucas	University of Southern California/ISI
David Manz	Pacific Northwest National Laboratory
Celeste Matarazzo	Lawrence Livermore National Laboratory
Jackson R. Mayo	Sandia National Laboratories
Masood Parvania	Arizona State University
Garrett Payer	ICF International
Sean Peisert	Lawrence Berkeley National Laboratory/UC Davis
Ali Pinar	Sandia National Laboratories
Thomas Potok	Oak Ridge National Laboratory
Stacy Prowell	Oak Ridge National Laboratory
Nageswara S.V. Rao	Oak Ridge National Laboratory
Eric Roman	Lawrence Berkeley National Laboratory
David Sarmanian	Integrity Consulting Solutions
Dylan Schmorrow	Soar Tech
Chris Strasburg	Ames Laboratory
V.S. Subrahmanian	University of Maryland
Vipin Swarup	MITRE

Brian Tierney	ESnet/Lawrence Berkeley National Laboratory
Von Welch	Indiana University

Appendix: Workshop Observers

Vergle Gipson	U.S. Department of Energy
Sandy Landsberg	U.S. Department of Energy
Larry Lanes	U.S. Department of Energy
Carolyn Lauzon	U.S. Department of Energy
Steven Lee	U.S. Department of Energy
Anita Nikolich	National Science Foundation

This work was supported by the Director, Office of Science, Office of Advanced Scientific Computing Research, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.