ASCR Cybersecurity for Scientific Computing Integrity — Research Pathways and Ideas Workshop

DOE Workshop Report

June 2–3, 2015
Gaithersburg, MD

U.S. DEPARTMENT OF ENERGY
Office of Science

BERKELEY LAB

15-CS-2135

ASCR Cybersecurity for Scientific Computing Integrity — Research Pathways and Ideas Workshop

DOE Workshop Report

June 2–3, 2015
Gaithersburg, MD

LBNL-191105 (2015)

Workshop Organizing Committee

Sean Peisert (Chair and Trustworthy Supercomputing Lead), Lawrence Berkeley National Laboratory and University of California, Davis

Thomas E. Potok (Knowledge and Analytics Lead), Oak Ridge National Laboratory

Todd Jones (Networks and Data Centers Lead), Sandia National Laboratories

DOE ASCR Point of Contact

Robinson Pino

Cover: Research necessary to assure scientific computing integrity in both current and future extreme-scale architectures will likely need to explore both novel computer security, software engineering, networking, and hardware/software-co-design techniques, as well as novel machine intelligence techniques, all while considering the way that scientists leverage computing systems to produce scientific results and how staff operate those systems. This cover is intended to reflect the interactions between all of these elements.

Preface

At the request of the U.S. Department of Energy's (DOE) Office of Science (SC) Advanced Scientific Computing Research (ASCR) program office, a workshop was held June 2–3, 2015, in Gaithersburg, MD, to identify potential long term (10 to +20 year) cybersecurity fundamental basic research and development challenges, strategies and roadmap facing future high performance computing (HPC), networks, data centers, and extreme-scale scientific user facilities. This workshop was a follow-on to the workshop held January 7–9, 2015, in Rockville, MD, that examined higher level ideas about 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 various 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 submission of abstracts relating to the topics discussed in the previous workshop report [1] and also from other ASCR reports, including "Abstract Machine Models and Proxy Architectures for Exascale Computing" [27], the DOE "Preliminary Conceptual Design for an Exascale Computing Initiative" [28], and the January 2015 machine learning workshop [29]. The workshop was also attended by several observers from DOE and other government agencies.

The workshop was divided into three topic areas: (1) Trustworthy Supercomputing, (2) Extreme-Scale Data, Knowledge, and Analytics for Understanding and Improving Cybersecurity, and (3) Trust within High-end Networking and Data Centers. Participants were divided into three corresponding teams based on the category of their abstracts. The workshop began with a series of talks from the program manager and workshop chair, followed by the leaders for each of the three topics and a representative of each of the four major DOE Office of Science Advanced Scientific Computing Research Facilities: the Argonne Leadership Computing Facility (ALCF), the Energy Sciences Network (ESnet), the National Energy Research Scientific Computing Center (NERSC), and the Oak Ridge Leadership Computing Facility (OLCF). The rest of the workshop consisted of topical breakout discussions and focused writing periods that produced much of this report.

Executive Summary

Contributors to the previous DOE ASCR Cybersecurity for Scientific Computing Integrity workshop, held in January 2015, defined several key reasons why the DOE should address issues surrounding scientific computing integrity, and moreover, provided numerous findings and recommendations regarding what research is necessary in order to understand how specifically those issues can be addressed [1] for example:

> "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 security and policy decisions. Moreover, the areas for which DOE is uniquely responsible, including energy, environment, and nuclear energy, all directly affect our nation's future economic prosperity and security. And in each case, scientific computing integrity is extremely important. ... it is vital that the results can ultimately be trusted ... 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 and infrastructure" [1].

This report builds on our previous report [1] with more specific details, aiming to define a long-term, 10- to +20-year fundamental basic research and development strategy and roadmap regarding scientific computing integrity facing future high performance computing (HPC) and scientific user facilities.

Specifically, this report describes potential research paths in three central areas: *Trustworthy Supercomputing, Extreme-Scale Data, Knowledge, and Analytics for Understanding and Improving Cybersecurity, and Trust within High-end Networking and Data Centers*. Within each of the three topic areas the report then addresses key technical areas. For each technical area, the report outlines the scope of the topic and potential milestones associated with research progress toward answering key questions raised in the topic.

More specifically, the workshop participants envision a future in science in the United States in which failures of scientific computing integrity in extreme-scale scientific computing systems, due to accident, natural fault, or malice, are as unlikely as one can expect computing failures to be within other high-assurance environments, such as computerized elements of military, avionics, and space systems [2,3]. The participants envision systems to support such a vision that leverage assured "co-designed" hardware and software computing, networking, and storage components. As many such components as possible should be a part of a process [4] and lifecycle in which they are specified, designed, implemented, configured, maintained, and operated leveraging foundational security principles [5 §I–VI]; with high-assurance methods [5 §VI, 6 Ch. 25–26, 7, 8]. Finally, when the resulting system is used for scientific computing purposes, deviations from scientific computing integrity should be "ideally provable, often measurable; and at minimum, possible to order and bound" [9].

Sometimes, methods for proving or measuring scientific computing integrity may not be not possible, because techniques are intractable or because using them causes undue interference with to the primary mission of producing science, such as those that generate excessively high energy consumption or computational performance overhead. In such cases, it is at least imperative that failure modes, including their possible causes and effects, are well understood, documented, and accounted for in hardware and software engineering design and implementation, and system operation.

Extreme-scale scientific computing systems must also contain means of generating evidence—highly detailed, and likely extremely high volume provenance data—to describe all aspects of the system pertaining to scientific computing integrity. Combined with new data ingress and storage systems that can capture and store both real-time computation data and also this provenance data, novel computing methods must be developed to analyze this vast amount of data and either demonstrate the integrity of scientific computing or provide indication of how integrity has failed and what the source of that failure was.

We envision that such systems must also enable and provide means for reproducibility of scientific computing results, taking all levels of the hardware and software stack into mind to account for subtle variations, such as compiler configuration, chip design, or some sort of non-determinism, that could affect scientific computing results.

Finally we envision an operational environment in which domain scientists are able to develop the software needed to perform scientific computing and system and network administrators are able to operate, configure, and maintain computing, storage, and network systems that contain as many of the same processes that support high-assurance scientific computing integrity as the development of these computing systems did to begin with.

Accomplishing all of these goals will not be easy. While at first glance much of it may "simply" seem like engineering, considerable research is necessary in numerous computer science, mathematics, engineering, and interdisciplinary domains. It is also vitally important to research methods for addressing competing and potentially contradicting elements to system design, including, first and foremost, maximizing the performance of scientific instruments to support its primary purpose: conducting science—and, not just "ordinary science" but extremely large-scale science of the type envisioned originally by Professor Ernest Orlando Lawrence using large teams of domain scientists and massive arrays of network-connected scientific instruments and other computational machinery. To this end, the workshop participants stress that scientific computing integrity should be viewed as an enabling technology, a core part of the scientific method, and a vital component of DOE's modern, open science. It should not, as computer security is often implemented, be implemented in a way that interferes with the science that must be performed. Other elements clearly include development cost, system usability, and energy needed to power extreme-scale scientific computing systems and any supporting systems performing analytics on high-resolution provenance data.

With all of these long-term goals in mind, the resulting product of the workshop, we believe, is a set of potential milestones and metrics that form a cohesive long-term research path of the type necessary to address needs surrounding scientific computing integrity at extreme-scale. This document describes that research roadmap and potential ideas for consideration.

1. Introduction

1.1 Problem 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 user facilities and computer systems, and of the scientific, engineering, and operational software and data that support its mission" [1].

Computer security is a significant challenge in virtually every place in which computers are used. Scientific, high-performance computing environments are no exception—security issues and incidents have occurred in such environments at least since the 1980s [10, 11]. In some sectors of society, a particularly valuable solution may simply be to disconnect critical systems from the Internet or perhaps any network [12]. But the requirements of open and international scientific computing environments preclude such a solution. Indeed, any such solution must support the fundamental open, international scientific functions required by scientific computing challenges. Furthermore, scientific computing integrity solutions must address failures not only from malicious actors—though such actors, including even in the context of hardware supply chain, software supply chain, and insider threats must be considered—but also includes faults involving flaws and bugs in the hardware, core software, operating system, libraries, compilers [13], and HPC infrastructure design and implementation errors. It must even include user error, errors in scientific codes and workflows, and natural faults such as bit flips due to cosmic rays or hardware component failures. Finally, solutions must also simply be usable by domain scientists and ultimately enable high-quality, *reproducible* science—a simple sounding term but a concept ultimately devilishly difficult—using the necessary computing machinery. Thus, these constraints help to define necessary research.

The contributors to the first DOE ASCR Cybersecurity for Scientific Computing Integrity workshop in January 2015 defined a key problem and recommended a solution:

> "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 energy, all directly affect our nation's future security and prosperity. And in each case, scientific computing integrity assurance is extremely important. ... it is vital that the results can ultimately be trusted ... 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. ...
>
> "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" [1].

That problem and the high-level solution from the first workshop was discussed by the contributors to the second DOE ASCR Cybersecurity for Scientific Computing Integrity workshop June 2-3, 2015.

The aim of this report, derived from discussions at the second workshop, is to define a long-term, 10- to +20-year fundamental basic research and development strategy and roadmap to address scientific computing integrity issues facing future high performance computing (HPC) and scientific user facilities, leveraging the findings discussed at the first ASCR Cybersecurity Workshop in January 2015, and presented in the subsequent report [1].

This report examines the research, design, development, implementation, deployment and application of computer security for scientific computing integrity technologies from data produced in experiments, simulations, emulations, transmitted, or at rest from sources or data centers. While excellent work

has been done on models and architectures for extreme-scale computing, much of this work has not addressed security and scientific computing integrity issues directly. Research and development into secure architectures and related analytics that integrate both security and integrity into an effective co-design process is very much needed, which is what this report attempts to describe.

1.2 Goals, Findings, and Objectives:

In our original report [1] from January 2015, a set of high-level recommendations were made about what the research and development needs are in order to enable scientific computing integrity and computer security, namely, *"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"* [1].

To address those recommendations with a more specific roadmap, this report describes research paths in three central areas: *Trustworthy Supercomputing; Extreme-Scale Data, Knowledge, and Analytics for Understanding and Improving Cybersecurity; and Trust within High-end Networking and Data Centers*. Within each of the three topic areas the report then addresses key technical areas. For each technical area, the report outlines the scope of the topic and potential milestones associated with research progress toward answering key questions raised in the topic, and means for measuring and evaluating success toward accomplishing those milestones.

In addition, the workshop contributors envision a future in science in the United States in which failures of scientific computing integrity in extreme-scale systems, due to accident, natural fault, or malice are as unlikely as one can expect computing failures to be within other high assurance environments, such as computerized elements of military, avionics, and space systems [2,3]. We envision systems to support such a vision that leverage assured "co-designed" hardware and software, networks, and storage systems. As many such components as possible should be a part of a process [4] and lifecycle in which they are specified, designed, implemented, configured, maintained, and operated leveraging foundational security principles [5 §I–VI] with high-assurance methods [5 §VI, 6 Ch. 25–26, 7, 8]. Finally, when such future systems are used for scientific computing purposes, any result deviations should be "ideally provable, often measurable; and at minimum, possible to order and bound" [9].

And, where such methods are not possible, due to intractable reasons or those contrary to the primary mission of producing science, such as those relating to energy consumption or computational performance overhead, or failure modes including their possible causes and effects are well understood, documented, and accounted for in hardware and software engineering design and implementation, and system operation.

Extreme-scale scientific computing systems must also contain means of generating evidence—highly detailed and likely extremely high-volume provenance data—to describe all aspects of the system pertaining to scientific computing integrity or its failure. Combined with new data ingress and storage systems that can capture and store the required provenance data, novel computing methods must be research and developed to analyze the vast amount of data and either demonstrate the integrity of scientific computing or provide effective indication of how integrity has failed and what were the source(s) of failures.

We envision that such systems must also enable and provide means for reproducibility of scientific computing results, taking all levels of the hardware and software stack into mind to account for subtle variations, such as compiler configuration, chip design, or some sort of non-determinism that could affect scientific computing results.

Finally, we envision an operational environment in which domain scientists are able to develop the software needed to perform extreme-scale scientific computing, and system and network administrators are able to operate, configure, and maintain computing systems, that contain as many the same processes that support high assurance scientific computing integrity as the development of these computing systems did to begin with.

Accomplishing all of these goals will not be easy. While at first glance much of it may "simply" seem like engineering, considerable research is necessary in numerous computer science, mathematics, engineering, and interdisciplinary domains. It is also vitally important to research means for addressing competing and potentially contradicting elements to system design, including, first and foremost, maximizing the performance of scientific instruments to support its primary purpose: conducting science—and, not just "ordinary science" but extremely large-scale science of the type envisioned originally by Professor Ernest Orlando Lawrence using large teams of domain scientists and massive arrays of network-connected scientific instruments and other computational machinery. To this end, the workshop participants stress that scientific computing integrity should be viewed as an enabling technology, a core part of the scientific method, and a vital component of DOE's modern, open science. It should not, as computer security is often implemented, be implemented in a way that interferes with the science that must be performed. Other elements include development cost, system usability, and energy needed to power extreme-scale scientific computing systems and any supporting systems performing analytics on high-resolution provenance data.

This is clearly an ambitious scope, not completely unlike the roadmap to build security enhancements into Multics [14, 15, 16, 17] for the Department of Defense back in the 1960s and 1970s.

We also know all too well that there is no "silver bullet" to solve these problems [18]. However, we believe it to be substantially more tractable than the issues that Multics sought to address, given more than 40 years of advances in computer science, computational capability [19], and a significantly more narrow scope in terms of scientific computing integrity, rather than security requirements that must also deeply consider deep issues pertaining to military-specific confidentiality needs, or the needs of the ordinary Internet more broadly [7, 8, 20, 21]. For example, perhaps not all computational components within extreme-scale scientific computing require general-purpose computing systems [9, 20, 21, 22, 23] and therefore carry with them their complexity and, consequently, their challenge of analyzing their properties relating to security and integrity. Other examples of highly assured systems, both old [24] and new [25], though smaller in scope, give hope for the prospects of future success with larger systems, as well. However, even research that enables nominally improved and more rigorous standards [26] that enable measurement of scientific computing integrity, in the context of scientific computing integrity, would be a substantial improvement over the current state of practice in high-performance, scientific computing.

1.3 Metrics of Success

As described in the previous report:

> "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" [1].

We note that overarching all of the technical research areas that we discuss in this document is a need to understand key scientific computing integrity properties for numerous applications of scientific computing in order to better characterize the constraints and need of potential solutions, and subsequently, metrics with which to measure success toward those goals. This is important to answer so that new architectural developments can be assessed. For example, there are a number of generally accepted key performance factors and metrics for HPC systems: power consumption, message-passing latency, floating-point operations per second (FLOPS) counts, cache coherency, addressability, etc.

Ultimately, the community must develop additional and effective scientific computing integrity-focused metrics that can be used to measure progress, to drive new innovation, and to assess the impact of new architectural changes on scientific computing integrity. This will be an ongoing aspect of all research in this effort, but we propose some high-level metrics throughout this document as examples.

1.4 Definitions and Organization of this Report

In this report, we describe potential research paths in three central areas, mapping to the topics of the breakout groups that were used at the June 2015 workshop. Those areas are: *Trustworthy Supercomputing; Extreme-Scale Data, Knowledge, and Analytics for Understanding and Improving Cybersecurity; and Trust within High-end Networking and Data Centers*. We note that there are overlapping aspects as well as symbiotic aspects between all three of these areas. This is both inevitable and intentional as the overlap suggests natural places where cross-disciplinary teams will need to work together to understand common problems and challenges and solve them together.

We largely define the scope of the *Trustworthy Supercomputing* topic in terms of the ways in which hardware and software need to be re-designed or enhanced, partially or fully, to assure and measure scientific computing integrity. This also includes the end-to-end nature of scientific computing workflows, as it includes the need to capture information from a variety of computing systems, and thus suggests re-design of any such system participating in that workflow.

We largely define the scope of the *Extreme-Scale Data, Knowledge, and Analytics for Understanding and Improving Cybersecurity* topic as a combination of: (1) the analysis of scientific computing integrity-related provenance data produced through experimental sources or computing systems, (2) the design and development of systems capable of collecting, representing, and analyzing that data, and (3) the simulation of supercomputers and scientific computing workflows.

Our definition of the scope of the *Trust within High-end Networking and Data Centers* topic is partially centered around the partial or full re-design and/or improvement of core networking protocols, systems, and technologies (separate from HPC), and also partially centered around the ways in which operational security personnel within data centers interface with computer systems, and thus the way in which systems are configured over time, and the way that security alerts and incidents are responded to.

Within each of the three topic areas we then address key technical areas. For each, we outline the scope of the technical area and potential milestones associated with each area.

At the end of the report we outline a few high level ideas on measuring progress down the path to successful research in this area.

2. Research Paths for Trustworthy Supercomputing

The first workshop report indicated a need to:

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 [1].

Given these recommendations, technical research areas appropriate for the *Trustworthy Supercomputing* R&D program are ones that bridge several technical domains. This is due to the recommendation that hardware and software be co-designed as much as possible to maximize the ability to perform key scientific computing integrity functions early on and in the most appropriate layer(s) of the hardware/software stack. In this section, we discuss aspects of this co-design that could be examined and a set of potential milestones intended to drive and measure progress toward scientific computing integrity goals.

2.1 Foundations of High-Integrity, Trustworthy Supercomputing

Today, computer users must place complete trust in the hardware and software stack of a computer, the system administrators that operate the computer, and typically anybody who has physical access to the computer. Similarly, an HPC scientific computing user is required to also implicitly trust the hardware, software libraries and codes they employ for mathematics, physics, output, and parallel and distributed programming.

In order to assure such systems and validate such presumptive trust, we seek a system, bootstrapped from first principles that are "ideally provable, often measurable; and at minimum, possible to order and bound" [9].

All scientific computing integrity rests on proper specification of the behavior of each component of the computing system, collectively and independently, as well as proper specification of the science, the scientific computation, and the tolerable variation in scientific computing integrity. Suitable specification languages are needed to enable proper and as complete and granular of specification and subsequent analysis of that specification as possible. Such languages must be analyzable by both the humans who create and review those specifications, as well as by the computers that must use automated algorithms to analyze specifications for internal consistency, adherence to fundamental properties, and avoidance of key design flaws.

Implementation is commonly separate from specification. In order to verify implementation, programming languages that are easier to verify are needed given the state space explosion problem of model checking and the computability limitations of automated theorem proving. Such languages should also aim to reduce the most pernicious programmer errors resulting in vulnerabilities and other implementation bugs in scientific parallel software. [30, 31] Such languages may even have reduced flexibility and functionality as a consequence of increased security without compromising scientific computing integrity. However, of particular importance to scientific computing integrity, these languages that reduce scientific computing integrity failures must not compromise the primary goals of computational throughput required in extreme-scale, parallel, scientific computing environments. Finally, such specification and programming languages must easily integrate with scientific workflow languages, including expected inputs/outputs.

However, as mentioned earlier, implementations are often separate from specifications. Thus, a final key property is that specification and programming languages must enable specifications, source code, and system configurations to be mapped to each other to verify consistency between the three.

That said, not all system components will be practical to formally verify or build in a formally verifiable way, due to cost, performance overhead, energy consumption, etc. Indeed, there are some fundamental tradeoffs that must be investigated to make decisions about the willingness of scientists to trust hardware and software components within extreme-scale HPC systems. Thus, a key question that reflects more near-term research might be: What system components can we really treat in a high assurance way, and if we did treat those components in a high assurance way, would that make a difference to scientific computing integrity?

For example, vulnerabilities in hardware designs such as the possibility of malicious circuitry being inserted by a chip designer, design tool, or at a fabrication facility have been

identified. To what degree should scientists trust that such extreme-scale HPC systems are absent of such vulnerabilities? If hardware cannot be trusted, techniques to mitigate the impact of embedded malicious logic should be explored to determine its utility in the context of extreme-scale scientific computing systems. On the other hand, if hardware can be trusted, it can also be used to provide trust anchors to increase the trustworthiness of software or to improve the efficiency of techniques that improve the integrity of software. Current examples of leveraging trusted hardware include encryption keys, trusted boot, and instruction set support for encryption and memory management support for virtualization and/or container domains. Potential future techniques include (but are not limited to) bounds checking, integer overflow exceptions, data tagging and provenance tracking, and hardware configuration checking techniques. Thus, perhaps a root of trust can be established and chaining mechanisms can reliably attest to the bootstrap configuration of the full scientific computing software stack. Another potential near-term research approach might be to assume the HPC hardware is trusted and consider the ways in which such an assumption can be leveraged to provide assured scientific computing integrity.

A great deal of the work described in this topic area relates to high-assurance engineering, and may seem on the surface not necessarily to include novel research. However, given the overriding specific context of scientific computing integrity, numerous open research questions exist, particularly questions about the applicability of many assurance techniques in an HPC environment and the trade-offs between cost, energy use, and performance if the techniques are implemented in extreme-scale machines. Research questions go well beyond these, however, and include fundamental issues in programing languages, compilers, and software engineering, as well. For example, what programming language properties and/or advances in compilers are needed in order to mitigate state space explosion problems when verifying software properties pertaining to scientific computing integrity? Or, what verification techniques can identify and eliminate properties entirely outside the specification (e.g., "weird machines") that could also impact scientific computing integrity?

In conclusion, we note that a number of these areas may represent early goals along the way, concurrent with longer-term research toward systems more specifically assured for scientific computing integrity. We do wish to note in passing, however, that looking to the future, neuromorphic and quantum computing are two potential future computational paradigms that each hold great promise for extreme-scale scientific computing. These architectures will need to be followed and researched as they become practical and effective. If so, scientific computing integrity research will need to be conducted on and for these architectures to understand the suitability for identifying and solving scientific computing integrity challenges particularly.

Potential Milestones

Potential milestones that we believe will help to lead to and measure research progress in this area include:

1. Identification of what system components can be treated in a high assurance way, and the effects on scientific computing integrity if we did treat those components in a high assurance way.
2. Identification of ways in which hardware roots of trust can be bootstrapped and leveraged to provide scientific computing integrity assurance, given the assumption of trusted HPC hardware.
3. Means of capturing specification (of specific properties) explicitly developed to capture scientific specifications, be analyzable, and be automatically and formally mapped to a programming language to verify correctness.
4. Identification of properties to be specified most relevant for HPC applications allowing different communities to focus on the specification and verification of those identified properties.
    a. Specification of key scientific drivers, built in this paradigm.
    b. Specification of key system components, built in this paradigm.
    c. Specification of complete scientific workflows, built in this paradigm.
5. Programming languages, frameworks, APIs, etc., explicitly developed to balance software performance, hardware power consumption, ease of use, and verifiability for extreme-scale parallel computers.
6. Design and implementation of next-generation operating systems that specifically target extreme-scale parallel computers consisting of distributed heterogeneous many-core processors. Inherent mechanisms need to be provided by the operating system to allow performance and security isolation without sacrificing power efficacy and throughput.
7. Implementation of core components specified, built, and verified using new language(s) and/or using diverse, redundant components.
8. Dependence graph of the current hardware/software stack (all libraries, modules, components, firmware, etc.).
9. End state: Development of a software/hardware architecture that provides the necessary primitives and structure to enable the full stack of security and integrity measures.

2.2 Isolation and Virtual Supercomputers

Much as in traditional computing environments, it is necessary in scientific computing to provide assurances that the actions of one user (and their software) do not affect the integrity of other system users. Memory management units are a classic

example of an enforcement mechanism that can be used to implement a variety of security and programming models. Hardware and software assurance, and co-design as described in the previous section, is a solution for this. By leveraging those assured components, however, it may be possible to increase functionality to create "virtual supercomputers" that are provisioned and instantiated for a specific job. This is similar to enforcing isolation within traditional dynamic "cloud computing" resources, though this concept focuses specifically on scientific, extreme-scale computing by providing users with a high-performance, parallel computing environment complete with low-latency, high-performance networks, wide area networks, co-processors, remote memory access and messaging middleware, run-time systems, and application software. Combining these properties with security isolation is a co-design challenge that builds upon the previous technical area and further requires all hardware components and their operating systems or firmware to either have high-assurance virtualization capabilities or to be assigned in their entirety to a single virtual supercomputer at a time and to have a high-assurance "reset" to a known good state between users.

There has been significant effort in the area of virtualization both in networks and system virtualization to provide isolation between users and processes in the area of cloud computing. This is currently extended to the network through the use of software-defined networking (SDN). The HPC community will need to adopt some of these approaches to provide isolation as well. Much of the focus of this topic, particularly in the near term, might leverage the techniques built as part of the previous technical area, although a parallel effort could assume that the software will not be secure. Secure software takes time and resources, and while new approaches may address these areas through the new language development, we will still have some remaining amount of legacy code.

Dynamic provisioning of resources in extreme-scale systems brings both new challenges and opportunities with respect to scientific computing integrity. For example, it may be harder to target a process or job that may be executed on resources that are determined dynamically, especially if the dynamically provisioned resources can be heterogeneous, which provides a sort of obfuscation for potential attackers. Additionally, virtualization can improve integrity by facilitating checkpointing and providing separation. On the other hand, virtualization also requires new techniques for certain capabilities, such as how to checkpoint and provide live migration for accelerators with separate memory spaces and how to share such resources without reducing the integrity of scientific computations. Additionally, allowing the use of dynamically provisioned resources adds complexity and volume to creating logs and performing audits and other sorts of monitoring. Finally, as we move toward extreme-scale computing, and given the corresponding drastic reductions in power consumption that will be required; the trend toward combining multiple components within a single component is likely to continue, such that it may be more difficult to devote an entire node or rack to one process, thus potentially increasing the risks for side channel attacks.

Potential Milestones:

Potential milestones that we believe will help to lead to and measure research progress in this area include:

1. Review existing isolation technologies that could be applied toward scientific computing integrity, including indications of their efficacy and gaps in existing technologies. Establish isolation metrics and performance overhead bounds.
2. Proof-of-concept virtually provisioned HPC "cloud" using a combination of efficient hardware support and potentially inefficient overlay mechanisms (such as encryption). Identifiy technology gaps to be filled to improve security and performance.
3. Virtual cluster co-design that provides quantified improvements to efficiency and security and includes multiple kinds of heterogeneous processing elements and considers both hardware and software techniques. This must become a sustained effort integrated into other co-design activities.
4. Work on introspection technologies for monitoring scientific computing integrity within job enclaves.

2.3 Automated Verification & Testing in Extreme-Scale Computing Environments

Reference workloads and benchmarks are used to test system performance and reliability. Similarly, formal verification has, since its inception, tested key properties of software. We lack a similar set of automated tests that validate scientific computing integrity properties of large, complex systems, as well as the ability to formally verify large groups of software and hardware components at scale. Existing techniques have limitations even in commodity computing, but these techniques need to be scaled to large, complex, distributed systems and specific architectures to find vulnerabilities and single points of failure, or regressions in the security assurances provided by systems.

Model checking has become a well-known approach for the automatic verification of program properties. The so called state explosion problem presents a significant challenge to model checking as the number of different program states can increase exponentially in an analysis. If the number of computed states exceeds the resource limitations of an analyzer, the desired property cannot be proven. Exhaustive analysis methods, which investigate all states of an application in some order, run out of resources for any system with an infinite number of

states. However, abstraction has shown to be a powerful tool to ameliorate the state explosion problem. The aim is to only inspect parts of the program that suffice to prove the desired set of properties. For example, counterexample-guided abstraction refinement serves as a concept to generate a coarse model of the analyzed program automatically. This allows one to better address the resource usage of a verifier, but still can require a substantial amount of resources.

Due to improvements in software verification algorithms, combined with increased computing power, verification techniques that a decade ago worked only for very simple programs are now applicable to real-world programs. In particular, if the national labs can utilize their own supercomputers for model checking and also use methods for abstraction refinement to address the state explosion problem, we can target much larger programs. How far can we push verification by utilizing supercomputers? If we are truly able to scale verification using supercomputers, it may also allow us to check HPC programs before they are run on supercomputers to ensure their integrity, similar to how we use supercomputers to develop supercomputers. Near-term research for verification of scientific software could and should certainly address this question for today's architectures while longer-term research should focus on future, extreme-scale architectures. Finally, functional testing of patches, updates, etc., could be made more rigorous. Could model checking used for this purpose identify examples that diverge between the unpatched and patched versions, and then more computationally intensive techniques, such as symbolic execution, be used to determine the effect of the patch or update?

Where programs cannot practically be verified, another verification technique might be the use of multi-version programming of scientific codes help to audit a scientific workflow, based on its inputs and outputs, and configuration for security/scientific computing integrity properties. For example, multiple versions or vendors of cores, codes, etc., all running in parallel could be built and a system bootstrapped from these [9] in order to perform the verification on an ongoing basis rather than just at compile time. This represents an opportunity for near-term research. Longer-term research could focus on the application of Byzantine consensus techniques for maintaining scientific computing integrity in extreme-scale systems.

Finally, an interesting part of the workshop interaction involved discussing the ways in which the scientific peer review and validation process, itself, can help discover and correct scientific errors. The scientific peer review process has demonstrated an ability to differentiate between significant and flawed work in the long term, though has been shown to be fairly noisy discriminator in the short term. While scientific peer review should undoubtedly continue to be a technique used for validating scientific programs, it is an open research question as to exactly how useful this technique is for assuring scientific computing integrity. There are other related, open questions, such as "What is known about the time frames necessary for one to state with confidence that the peer review/citation history is sufficient to conclude something meaningful about a particular body of work?" Additionally, how might more complete information on the experiments (provided it can be delivered in a form that is useable by the review process) help improve the efficacy of the peer review process? How do we leverage the peer review/scientific self-correcting issue? How can we assist the peer review process? Where does peer review fail? Does validation of provenance data at various points, such as against physics models, conservation laws, and independent results also provide useful insight?

In this technical area, we emphasize five, broad categories:

- Development of program verification techniques that can meaningfully leverage HPC capabilities to perform the verification (e.g., via parallelism).
- Use of diverse redundancy of scientific codes to evaluate degree of independence of the system on current systems.
- Large-scale, automated "stress testing" of HPC environments (particularly inputs to HPC programs) to evaluate accuracy.
- Formal verification, including model checking, theorem proving, and fault tree analysis at very large scales (and leveraging HPC platforms themselves to do so), including model checking of entire scientific workflows, including HPC hardware, systems software, scientific instruments, and network systems. This will help to find single points of failure and thus the places that are the most vulnerable to integrity failure.
- Analysis of application of the scientific method/process itself to evaluating scientific computing integrity failures.

We note that this technical area leverages the language and compiler design elements from the first technical area.

Potential Milestones

Potential milestones that we believe will help to lead to and measure research progress in this area include:

1. Fault tree analysis for a set of key extreme-scale science drivers for a preliminary extreme-scale system.
2. Experimental analysis of multi-version programmed science applications that demonstrate deviations in scientific computing integrity for a set of key extreme-scale science drivers.
3. Development of models of how accurate the scientific peer review/citation impact measurements (e.g., impact factor, *h*-index) are at identifying flawed experiments at various time scales.
4. Methodology for long-term evaluation of bodies of work taking into account automated verification, experimental replication, and analysis of the response of the scientific community.

5. Method to develop and update necessary specifications from the imperfect starting requirements with traceability supported (so code does not drift from specifications without the specification also changing) via automated tools.
6. Scalable formal evaluation of sets of individual but significantly large components (whole OS, large science apps, individual hardware components, etc.). This also requires that components and their interfaces are all clearly defined.
7. Ability to derive composable specifications from components and to reason about their potential and actual compositions, enabling the ability to reason about the interaction of large sets of such components.
8. Implementations of diverse, redundant components of key system components at single points of failure based on fault-tree analysis (FTA).
9. Characterization of key elements of the value of the scientific peer review and evaluation process to correct errors, including indications of time frames, efficacy, means for improvement, failure modes, etc.
10. End State: ability to perform parallel, scalable formal verification (model checking, automated theorem proving, and fault tree analysis) across the entire hardware/software/scientific stack.

2.4 Scientific Computing Evidence Records—Provenance of Extreme-Scale Scientific Workflows

Not all properties of scientific software can be proven and evaluated at compile time. Some things must be done at runtime for computability, cost, or efficiency reasons. Moreover, we must capture not only elements that cannot be proven but elements that provide solid evidence of integrity or lack thereof, as well as scientific computing reproducibility. Measurement and recording of the context in which results were achieved is vital to the latter. This suggests data capture and analysis for provenance and configuration in either near-real time, post hoc, or both. Co-design could include several layers of the software-hardware stack working in concert to do this. Given this, however, what log and configuration data should be captured, and where in the process to provide evidence of scientific computing integrity? What mechanisms could be useful to enable data to be "born" with digital signatures from the outset? How do we specify and analyze a scientific workflow (inputs/outputs) configuration for security/scientific computing integrity? How do we perform ultra-high resolution capture, reproducibility, and analysis of entire, distributed hardware and software environments?

A research program that builds on the results and output of existing scientific workflows to help influence or develop cybersecurity-minded workflows that target trusted computers that can exploit heterogeneous hardware and improve or ensure scientific computing integrity could make a significant impact on the productivity of trusted supercomputing in the extreme-scale computing era while improving the integrity of science produced on those platforms.

This research area, and its adaptation to existing systems, has challenges across numerous fronts, including impact on computational efficiency, hardware power use, and insight into scientific computing integrity. For example, audit data is often quite opaque—methods to reduce the semantic gap between the provenance data and the executing workflows should be explored and incorporated directly into the software stack. New security features needed at the runtime layer could inform a new OS design and in turn require changes to the compiler, linker, and even the hardware to accommodate these features. In parallel, numerous open research questions involving extreme-scale architecture arise as well, such as "With the colossal number of processors available, how can spare cores be used to run domain-specific data checking algorithms?" and "Are low-impact, out-of-band data operations for integrity checking or provenance data collection purposes practical on extreme-scale systems?"

The development and exploitation of scientific workflows may provide the opportunity to address the two extreme-scale scientific computing challenges of how to improve programmability of complex heterogeneous architectures and scientific computing integrity. Because workflows often implement restricted or specialized computational models, it becomes more feasible to perform analysis on them than is possible with more general-purpose computing. Such analysis can be used to enable reproducible scientific computing, to optimize for performance, generate code that targets heterogeneous computing elements, and to check for scientific computing integrity.

Potential Milestones:

Potential milestones that we believe will help to lead to and measure research progress in this area include:

1. Working with domain scientists, identify reproducible qualities of key scientific drivers, including gap analysis of existing workflow tools, to capture these qualities.
2. Identification of minimal information necessary to reproduce scientific results.
3. Perhaps first for software configuration, then for hardware configuration.
4. Development of capabilities (hardware and software) with acceptable performance impact to capture and store the necessary runtime and state information.
    a. Perhaps first for software configuration, and scientific workflow components, and then for hardware.
5. Analysis capabilities to prove the source of failures or the absence of errors in the face of claims, evidence, or lack thereof.

6. Human factors experiments to determine how provenance information can best be useful to the peer review process specifically and the scientific method more broadly.

3. Research Path for Extreme-Scale Intelligent Data, Analytics, and Knowledge (ESI-DAK)

Systems and techniques will need to be developed to autonomously and intelligently capture, analyze, and correlate actionable information from extremely high volumes of provenance information and other data related to scientific computing integrity to provide supportive and decisive evidence about scientific computing integrity—even in high assurance systems [32]—or determine where and how it has failed. This also includes the end-to-end nature of scientific computing workflows mentioned earlier. The scope of the *Extreme-Scale Data, Knowledge, and Analytics for Understanding and Improving Cybersecurity* topic area is defined as a combination of: (1) the design and development of systems capable of collecting, representing, correlating, and analyzing massive amounts of data related to scientific computing integrity assurance; and (2) the simulation of supercomputers and scientific computing workflows to assess risks and explore mitigation strategies, including the need to examine the real-time control flow of the software itself. This requires the ability to autonomously and dynamically analyze running software, and to detect potential threats within very large simulation codes during the computing phase.

The January 2015 *ASCR Cybersecurity for Scientific Computing Integrity* report [1] recommended:

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:

1. ***developing an analysis framework*** capable of collecting scientific computing integrity data at an unprecedented scale from multiple sources that together represent the system under study to enable adaptive, streaming analysis for monitoring and maintaining scientific computing integrity;
2. ***developing means of 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;
3. ***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
4. ***developing new methods for meaningful risk measures*** and threat measures of HPC integrity.

Most notable among these recommendations is the emphasis on real-time, in situ analysis of extremely large volumes of scientific computing integrity and provenance data. This is due to the fact that monitoring of extreme-scale-class systems will generate extremely large amounts of data which cannot easily be stored offline for later analysis. This is particularly applicable to monitoring HPC systems for executing code integrity, where it simply may not be viable to store all the relevant application-level monitoring data. Thus, analysis will need to be performed *in situ* and in real-time. This means that the methods will need to be extremely fast with high throughput, while retaining accuracy, and being sufficiently lightweight to not have an adverse impact on the scientific computation being monitored. In contrast, most current machine learning techniques are very heavyweight and thus not suitable for this problem. Substantial research is needed in statistical and machine learning approaches that are sufficiently lightweight to deal with high-volume, real-time streaming data, as well as the analytic infrastructure surrounding those learning approaches. In particular, the focus should be on the domain of scientific computing integrity, which should enable more tailored approaches.

We identify three main research areas that address the above recommendations as follows:

1. ***Real-Time, In Situ, Scalable Analytics, including Machine Learning and Neuromorphic Computing Approaches,*** combining recommendations 1, 2, 3, and 4.
2. ***System Metrics and Simulations, combining*** recommendations 3 and 4.

3.1 Real-Time, In Situ, Scalable Analytics Framework

A key research area involves understanding how to develop an analysis framework capable of detecting abnormalities and root causes of loss of scientific computing integrity, using data from multiple sources that collectively represent the system under study. Such sources include data or data products of the scientific computation itself; cryptographic hashes, checksums, or other representations of that data; data about the state of other activities transpiring on the system and network; and extremely fine-grained provenance data containing representations of hardware and software used, actions within the scientific workflow, effects of system and network behavior on the scientific computation or its output, non-deterministic events, and more. As we discuss in the next section, scientific computing integrity-related data will also need to compare these analyses with the outputs from various simulations.

Given the massive amounts of data produced by an HPC system, it is likely not feasible to store all of the information for later analysis. Compounding the problem, the velocity and volume of data will continue to increase at prodigious rates. Given this, while data collection and management approaches exist in other ASCR programs, this workshop, as well as the recent machine learning workshop [29], recommended that new techniques be developed to focus specifically on data collection and management techniques and analytic methods that may be specific to the challenges involved in collecting and analyzing data needed to prove scientific computing integrity or determine where and how it has failed.

Extreme-Scale Scientific Computing Integrity and Provenance Data Collection and Representation: As a first step, a capability is needed to collect and represent scientific computing integrity data in situ and at an unprecedented scale. This data will be from multiple sources, and collectively will represent the extreme-scale computing systems under study to enable analysis for monitoring and maintaining scientific computing integrity, including reasoning about causes of integrity failures. New data representation formats that are able to work with a broad range of data formats, in very high volumes and with very fast access times, may also need to be developed. These representations will also have to potentially capture vital temporal relationships that might illuminate key distinctions between one computation and another, due to non-deterministic concurrency differences, for example.

Extreme-Scale Scientific Computing Integrity and Provenance Data Analytics: Beyond means of collecting and storing integrity-related and other provenance data, we must develop very fast ways to analyze data for integrity-related problems, detect when those problems occur, and perform real-time responses to ameliorate or limit integrity issues where feasible. Thus, analytics must also provide an interpretation of the semantic meaning of the provenance data such that anomalies are actionable by either human operators, automated systems, or some combination of the two. To accomplish all this, new extreme-scale statistical, machine learning, and graph-theoretic techniques are likely required. Given the limited ability to store massive volumes of scientific computing integrity and provenance data, analysis will also need to be performed on the fly, in situ, with near-linear time complexity for real-time performance.

Data Reduction: Given the sheer volumes of data generated as part of scientific computing, and also specifically collected as required for provenance analysis, we recommend developing methods to quickly and accurately parse log data separating the signal from the noise. A major problem with large log data is the need to clean the data and then fuse the data based on time or some other feature. Given the complexities of extreme-scale HPC, knowing which data is relevant to the scientific computing integrity, a priori, is a research challenge since some data will inevitably need to be discarded. For example, when can one throw out sensor data after analyzing it, given that a sensor itself may be faulty? Despite this significant challenge, to accomplish this means that either methods for real-time analysis of the data will need to be developed, or alternatively, adaptive sampling methods would be needed to quickly expand and steer the sampling of data when a loss of scientific computing integrity is identified or suspected. Research should certainly focus on interpretable dimensionality reduction methods, as well as methods to identify recurrent patterns that represent normal system behavior as opposed to abnormal behavior. Further, research in extreme-scale data fusion and reduction as they relate to scientific computing integrity provenance data are likely to be useful.

Statistics and Machine Learning: Real-time anomaly detection tools for massive streaming data that correspond to a mixture of scientific computations, byproducts of those computations, security-related logs to HPC machines, sensor data and system-logs from HPC systems, as well as outputs of in situ data analysis from the simulation to detect malicious or unusual behavior, are necessary. This requires anomaly detection that can work with different forms, modalities, and fidelities of data from numerous sources (simulation, instrumentation of HPC, and logs). Extreme-scale parallelism must be fully exploited in both model building and model querying. Additionally, there need to be new methods developed for rule-based statistical methods, outlier detection, and probabilistic models. Additionally, we note that the data produced by extreme-scale systems will hold both spatial and temporal locality. Since deep learning methods have excelled in applications that exhibit these characteristics (e.g., image classification and speech recognition), deep learning methods should be developed to identify malicious action within these extreme-scale systems. Developing such a system will require addressing data management, data selection, and training method challenges resulting both from the size and unique characteristics of the data. Possible uses for these models could be anomaly detection, attack recognition, and generating compressed representations of the data.

Graph Analytics: Computer security data, and therefore likely also scientific computing integrity provenance data, typically represents highly connected data. One way to capture interdependencies between security-relevant events is through a graph. Storing data in the form of graphs will enable efficient querying. However, in the context of extreme scale, these graphs will become unwieldy and will need to be stored in a way that is amenable to different types of analysis. Data representations for scientific computing integrity would require graph models that are far more complex than what has been done before. One could envision a case where data from multiple sources, applications, system sensors and users all being represented, streaming in and being analyzed at the same time. The data structures needed for such a model have to be dynamic, scalable, and able to handle multiple, different attributes on vertices and edges. Graph-based methods for this data must also take into account the topology of the (event) network, which is complementary to many statistical learning-based methods that assume independent and identical

distribution of events. As with statistical learning, extreme-scale parallelism has to be fully exploited in both model building and model querying for graphs as well.

Graph models have traditionally been applied to a variety of applications such as scientific simulations, representing node architecture of supercomputers, social interaction, and road networks. Traditionally, in all of these problems, the graphs are treated as representation of static data. With the right data structures, graph-based anomaly detection and pattern detection will lend information on the integrity of the scientific data in a scalable manner.

Graph models of streaming data are interesting because they support a compact, discrete representation and can be used to model behavioral features associated with computer security threats. Dynamic graph models provide a promising approach for modeling data movement, network traffic, and application interactions in extreme-scale systems. Research challenges include (a) developing infinite-stream streaming algorithms (which must analyze data in a single pass), (b) storing partial patterns to efficiently support complex pattern matching, and (c) co-design of extreme-scale systems to support scalable streaming techniques (including both custom hardware and system software).

Finally, as mentioned earlier, real-time scientific computing integrity analytics will be required not only to process data from extreme-scale computations but also from simulations so that the results can be compared. And the scale of data from extreme-scale scientific computing simulations is likely to be massive. Lightweight, application-specific graph models are needed that can be coupled with extreme-scale systems in situ or ex situ analytics for clustering and inference problems for pattern and anomaly detection. Data at this scale could also be compressed or sampled. For example, sketching based on sampling and random projections has been very useful in linear algebra and could help reduce the scale of data in memory and/or reduce the cost of the analytics. Techniques such as linear algebra-based graph analysis tools and partitioning tools for sparse, irregular, temporal data will play an important role as well.

Integrity Data Analytics with Adversarial Robustness: Compounding the issue of anomaly detection is the adversary who understands the analytics and may seek to subvert our analyses. Specifically, our adversaries may have the knowledge and know-how to manipulate our analyses to evade detection of malicious activity or, potentially worse, to waste valuable human resources by causing false alarms that an analyst must investigate. For these reasons, "adversarial robustness" of the methods is a crucial feature because good intruders have tools to blend in as a regular user.

Current and Future HPC Architectures: The analysis of large sets of scientific computing integrity data can and should be analyzed directly on DOE's various HPC platforms, including those at NERSC, ALCF, and OLCF. A current trend in large data analytics is the use of specialized hardware such as large shared memory computers and/or systems with large-scale flash-based storages. In situ analysis of scientific computing data will require scalable data analytics in the supercomputers that are built for traditional scientific simulations. Keeping with the trend of the predicted extreme-scale hardware, the analytics also have to be thread scalable and performance portable for the different accelerator hardware in extreme-scale systems. Both scalable analytics in HPC platforms and on-node acceleration of these analytics are key components of successful analysis of scientific computing integrity data.

Potential Milestones

Potential milestones that we believe will help to lead to and measure research progress in this area include:

1. Methods for performing appropriate data dimensionality reduction on scientific computing integrity-relevant provenance data.
2. Methods for fusing extreme-scale data of the type and scale to collect for scientific computing integrity analysis.
3. Graph-based and statistical/machine-learning techniques that capture multimodal data attributes, leverage extreme-scale parallelism, are useful for evaluating scientific computing integrity-related data that take into account the topology of the (event) network, and present analytic interpretations of scientific computing integrity data such that anomalies are actionable by human operators.
4. Methods for developing analytic interpretation of provenance data so that causes are detectable and anomalies are actionable by automated response systems.
5. Statistical, machine learning, and neuromorphic computing-based detection methods useful for evaluating scientific computing integrity-related data robust to adversarial attack.
6. Methods for performing full-stack scientific computing integrity data analytics, in situ and in real time on scientific computing integrity data.
7. As architectures evolve in the future, statistical, machine learning, and graph theoretic methods useful for scientific computing integrity that leverage performance of neuromorphic and/or quantum computing architectures.

3.2 System Integrity Metrics and Simulations

Developing capabilities to model, quantify, and manage extreme-scale performance to allow extreme-scale computing users and system operators to effectively manage the tradeoffs between scientific throughput and scientific computing integrity is a vital but complex research problem. We therefore recommend

development of metrics and models to understand risk and the relationship between computer performance and scientific computing integrity. Proving the complete integrity of a scientific computation is likely a very computationally intensive process and doing so would impede the progress of science. Considering the fact that power and energy requirements of extreme-scale systems is a fundamental limitation, expressing security as a function of energy consumption (or with power constraint) will also be critical. Further, it would be useful to consider metrics that evaluate the trade-off between the integrity of the computation and the performance penalty.

To accomplish these goals, it will be necessary to model and simulate numerous elements of the process, including:

Scientific Computing Workflow Models: It is vital to study and model how a scientist interacts with an HPC environment from a security standpoint to look for potential bottlenecks, including those caused by security protocols and potential security vulnerabilities.

Vulnerability Models: The HPC community drives innovation in both software and hardware and are often the first adopters of custom technologies. This puts the community out on the bleeding edge of technology where security is often overlooked for features or simply functionality. The ability to model interdependencies between software and hardware components in a scalable manner is therefore an important research topic. Current methods for modeling, such as attack graphs and Bayesian attack graphs, are fundamentally limited in their ability to scale to the levels of interest to DOE. Multi-level methods can be a potential approach for scaling attack graphs by coarsening the graph at several levels and accumulating vulnerabilities and exploits at those levels. However, tradeoffs between accuracy and scalability need further study. Therefore, the documentation of threats specific to HPC, as well as continuous research directly linked to the security of new technologies as they are proposed for inclusion in the HPC system stack are all needed. In some cases, threats might even be simply the most common flaws made in developing software for extreme-scale systems.

Failure Models: Identifying failures of scientific computing integrity, including that resulting from malicious behavior, requires labeled data, which can be generated from previously seen threats and red/blue team exercises. Though history and experience have shown this to be an incomplete solution at best, it can provide useful information. Although one challenge with statistics and machine learning for computer security is the high dimensionality of the data, the lack of training examples and the need for retraining when domains change is an additional challenge. Thus, methods need to be developed that can update and augment the training data for more accurate classification. In addition, methods are needed that can automatically correct the model due to drift or domain change. A key challenge is developing a framework that supports the integration of user feedback on integrity, and thereby supports the overall improvement of integrity within an extreme-scale system.

Metrics for Automated Response: Coupled with the idea of real-time processing is automated response, the ability of systems to automatically react to an anomaly or problem without human input. The motivation for this is two-fold. First, human operators are a scarce resource, and will form a critical bottleneck if they have to determine what to do with every anomaly. Second, having to wait for a human operator to react means that the damage may already be done. What is needed is to stop the attack before it can do harm. Of course, automated response is a tricky issue, since false positives can result in loss of service and other issues.

Moreover, it is important that development of automated responses take into account how the users might react to the methods. For example, if changes are made to a user's workflow that make the process more complicated, confusing, or slower, the user might resist the changes—even if the goal of the changes is to improve scientific computing integrity—if the benefits are not articulated to the user. Similarly, if changes are made to enhance defense or detection analysis and the adversarial response is not considered, it could place the scientific research infrastructure at greater risk. If the adversary knows the sampling rates or threshold for the analytics, they could simply outwait or fall below the detection levels. This is not "security through obscurity"—rather, it is the acknowledgment that users (benevolent and malicious) will adapt to a response and long-term security and stability of the system will be influenced by their behavior, as much by any defender actions. Consideration that the adversary will adapt should also include countermeasure detection. Analytics developed should look for data that may be skewing anomaly detection data.

Simulation of HPC Performance with Scientific Computing Integrity: Using models of scientific computing workflows, vulnerabilities, automated response, and the HPC systems themselves, one can run simulations of cause and effect to better understand and refine predictions and analytics about potential loss of scientific computing integrity, and also to test mitigation strategies. For example, modeling and simulation are currently being used to predict networking protocol performance and software run-time performance on various hardware platforms. Modeling of computational patterns has also been used to successfully "fingerprint" the types of programs running on high-performance computing systems via their runtime communication patterns. Tools used for these types of efforts may be extended to include methods and analysis for assuring scientific computing integrity. The conceptual components of a model of computational infrastructure for both HPC systems and general communication networks are computer nodes, networks that connect nodes, software stacks on the nodes, simulation applications, and end-user scientists involved in the workflow. An integrated modeling capability for all these components could be used for comprehensive evaluation of

future systems that includes sensors, tools, and analytics for data integrity.

Potential Milestones

Potential milestones that we believe will help to lead to and measure research progress in this area include:

1. Develop models of key scientific computing workflows .
2. Develop models of key vulnerabilities in scientific computing environments.
3. Develop failure models to scientific computing integrity.
4. Develop metrics of defining the criticality levels of any identified vulnerabilities must be developed.
5. Develop metrics and methods that enable measuring, analyzing, and characterizing trade-offs between resolution of understanding and maintaining scientific computing integrity in comparison to performance overhead and energy (power) consumption.
6. Develop metrics that can inform automated response to maintain or prevent further loss of scientific computing integrity.
7. Develop full-stack hardware/software simulations of HPC systems and end-to-end scientific computing workflows that can enable comprehensive testing and evaluations of scientific computing integrity within such environments.

4. Research Path for Trust within Open, High-End Networking and Data Centers

\As discussed in the two previous topic areas, there are several operational requirements for scientific data integrity in data centers, including both the ability to detect loss of data integrity, whether due to natural, accidental, or malicious causes; and recover from the loss of data integrity. Given those requirements and the assumptions of successful implementations via the *Trustworthy Supercomputing* topic area and associated analytics of supporting integrity-related data, the questions remain as to how operational security personnel will interface with and act on these assumptions and related data, and also how these elements will interface with underlying networking technologies.

Simultaneously, it is essential to minimize the interference (e.g., due to performance overhead) with the core science being performed. We also need to minimize the burden on individual users, including administrators and scientists, to do the "right thing" with regard to security and scientific computing integrity. Thus, a system that brings core scientific computing integrity capabilities together in an operational environment should aspire to assure such integrity via visibility, usability, and evidence. It should allow us to detect, locate, and attribute any modifications made to data produced by an extreme-scale computation, or to results derived from that data, regardless of when and where that modification occurs. A researcher should be able to readily determine the source, provenance, mutation history, location, and status of any and all data associated with their research computations. A researcher should also be able to reproduce scientific computing results.

The first workshop report indicated a need to:

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.

4.1 Trust within Open, High-End Data Centers

The end-to-end processes that produce computational science results are large, complex and afford many opportunities for data to become accidentally or intentionally corrupted or lost. Data center services and networks form a layer that links high-performance computing resources, specialty instruments, other experimental data sources, storage, and the international scientific community. In particular, this layer also represents the connection between computing systems and human operators who may take actions related to scientific computing integrity based on insights given by real-time system analytics. Thus, any process for safeguarding the integrity of scientific computing must consider what actions operators, system administrators, and scientists might want or need to take.

Methods for representing scientific computing integrity status, recommended actions (such as rollback and recovery), and the impact of those actions (e.g., loss of *n* number of days of work)

are essential aspects of this topic. Also important are methods for integrating these techniques both within DOE scientific instruments themselves (e.g., light sources and particle accelerators) as well as providing a framework that DOE researchers that they can use with systems outside the DOE but nonetheless commonly used by DOE researchers, such as the Large Hadron Collider and various computer-controlled telescopes.

Potential Milestones

Potential milestones that we believe will help to lead to and measure research progress in this area include:

1. Develop an evidence-based risk model based on scientific computing integrity needs, describing the nature and scope of threats to scientific computing integrity to inform efficient risk management processes at the DOE labs and characterize desirable technical responses, either manual or automated.
2. Develop interactive systems, usable by both scientists and administrators, for representing scientific computing integrity, and provenance state and supporting analytics surrounding that data.
3. Develop high-performance means to develop an interactive system, usable by both scientists and administrators, for recovering from loss of data integrity to last known uncorrupted state, and for reproducing questionable computational results, based on provenance information.
4. Develop a scientific computing integrity workflow model such that all DOE ASCR computing data is associated with an identifiable operational computer security entity.
5. Expand the previous milestone to include computing systems and instruments outside of DOE's computing environments.
6. Incorporate scientific computing integrity features into "data at rest" storage systems, such as a distributed, replicated, versioned file system appropriate for use in extreme-scale environments.
7. Research and develop federated identity/authorization solutions for the DOE ASCR complex and open science partners that includes notions of transitive and brokered trust relationships.

4.2 Trust within Open, High-End Networking Environments

Increased use of encryption, alongside new network technologies such as software-defined networking, packet optical networking, photonic switching, and other hybrid and next-generation equipment, poses challenges to or will completely obsolesce current network security analysis practices. However, the ability to observe and analyze network traffic in real-time will remain essential for monitoring open science computing environments.

The following questions describe the major information security research questions over the next two decades. In the section that follows, we describe some potential shorter-term research projects and milestones:

Security in the Context of Encrypted Network Traffic: Encryption is quickly becoming ubiquitous not just in banking environments but even on common news and search engine web sites. Thus it is necessary to understand how anomaly and attack detection continue to be effective in a world of ubiquitous encryption at multiple layers. For example, it is necessary to understand if endpoint analytics are able to replace network monitoring while still preserving privacy and confidentiality. It is also necessary to understand how encrypted transactions can be successfully monitored while continuing to allow for confidentiality and whether events, such as exfiltration of data, can be identified in real time without having visibility into the data stream contents. Correspondingly, it is vital to incorporate semantic information bearing on scientific computing integrity from sources other than just the network itself—for example, end hosts, authentication infrastructure, external intelligence, and databases. It must also be necessary to integrate this information at very high speed/low latency and with substantial privacy protections. As discussed in the other topic areas, it is likely also vital to incorporate automated responses at fine-granular semantic levels in both hosts and networks, for example, through software-defined networking. In parallel to the need to operate on encrypted data, it is necessary to determine if host analytics can inform data reduction techniques to allow only a small fraction of full network traces to be collected and stored, while still providing adequately rich inputs to integrity data analytics.

Software-Defined Networking: As networks become more programmable and virtual via software-defined networking (SDN) techniques, new vulnerabilities may be introduced. So it is necessary to determine how to maintain the trust and integrity of the network and the data that moves in and out of it. For example, it is important to determine how the trust-based nature of the Internet resource exchanges (e.g. BGP) be mapped into a software-defined networking environment where resource authority is largely based on the trust model. Can resources be translated, authenticated, and verified in near-real time to a level that can be deployed analogous to how BGP4 exchanges route prefixes? At the same time that SDN introduces new challenges, SDN may also introduce opportunities as well [33], such as leveraging new trust models, scientific computing integrity-related provenance data collection, etc. Thus the introduction and application of SDN in scientific computing remains a significant research area with regard to scientific computing integrity.

Quantum Networking: Quantum networking describes a set of techniques, many under active research, for harnessing

quantum-mechanical effects to achieve new capabilities in the transfer of information. Two main applications for quantum networking have been discussed widely. The first is the ability to transfer quantum (non-classical) state between geographically-distant quantum computers. Such transfer might be required in a scenario where quantum computers need to be coupled together in order to achieve a required outcome. Some researchers have even imagined an architecture and protocols for a future quantum Internet, interconnecting a globally distributed set of future quantum computers. The second application involves provably secure information transfer, using quantum key distribution (QKD). Forms of QKD have already been implemented in a number of testbeds and research networks. The overall aim is to exploit quantum effects to render in-transit data manipulation impossible, or statistically infeasible, and enhance information transfer integrity. QKD appears conceptually promising as a part of a framework that might help to enable scientific computing integrity assurance. However, the viability of other quantum networking techniques, such as the "Quantum Internet" concept, or quantum computing more generally, remain longer-term, open research questions.

Integrating and Employing Real-Time Analytics of High-Throughput Data Streams in Automated Responses: It seems clear that the number of distributed sensors analyzing network performance and scientific computing integrity are likely to be useful in some fashion, but it is unclear how we will best be able to actually benefit from rapid real-time data analytics across myriad data streams. Open questions include how we can detect things that are not otherwise immediately observable, and how massive data correlation can play a significant role here? Can distributed analytics be used to aid in accomplishing this (e.g., using route views, perfSONAR-style distributed sensors, or even the RIPE Atlas measurement framework)?

"Clean-Slate" or "Greenfield" Internet Architectures: The Internet was not designed for security. What is the architecture that will enable trust in scientific computing integrity in the Internet of the future? Will that architecture be a "clean slate" or "greenfield," unconstrained by previous design decisions? Can the Internet as it exists today be modified and improved in such a way that it can reach a level of adaptability that can adjust for security incidents, needs, requirements and events similar to how routing in layer 3 of the OSI stack heals when resources change state? Can collapsed platforms such as packet optical routers and other CPU-rich devices aid in the goal by using more DPI and flow-based analytics in a programmatic manner (SDN or self-[re]provisioning networks)? Information-centric (ICN), content-centric (CCN), and named data networking (NDN) are all concepts related to a particular aspect of future Internet architectures (indeed have, in some cases, been even developed as part of NSF's Future Internet Architectures) program, and represent a tremendous opportunity for advancing security in networked computing environments. One key reason for this is that is that ICN, CCN, and NDN data objects are digitally signed "at birth," thereby providing a key source for assuring integrity of data at its instantiation. However, ICN, CCN, and NDN have not been integrated into popular use or even custom HPC or scientific environments. Nevertheless, all represent specific examples in which future networking and computing architectures could make substantial advances over current technologies and should certainly be examined for their applicability to HPC environments and scientific computing workflows. However, it is vital that "greenfield" designs be more broadly considered as part of a similar redesign/co-design of the trustworthy computing hardware and software stack.

Potential Milestones

Potential milestones that we believe will help to lead to and measure research progress in this area include:

1. Develop means of real-time monitoring scientific computing integrity aspects related to networks in the face of encrypted traffic without compromising confidentiality or user privacy.
2. Develop mappings between Internet resource exchanges (e.g., BGP routes) based on trust models.
3. Develop means to incorporate automated, real-time analytics into triggers to changes in the network environment, e.g., via SDN, that incorporate analytic input from the ES-DAK topic area to maintain or prevent further loss of scientific computing integrity.
4. Conduct initial research in applications of QKD and other quantum networking techniques to scientific computing environments.
5. Conduct initial research in greenfield networking architectures that can be modified to support scientific computing integrity.
6. Continue research in greenfield networking architectures likely including ICN, CCN, and NDN that can be modified to support scientific computing integrity.
7. Implementation of novel high-integrity networking techniques, as determined feasible, based on research in greenfield networking architectures that can be modified to support scientific computing integrity.

5. Measuring Progress on the Path to Success

Ultimately, the real test is measuring broad progress toward improved scientific computing integrity by comparing results on current systems with results from future systems and comparing deviation from acceptable ranges of computing integrity as established by domain scientists, and weighing these improvements against external costs such as performance overhead and energy consumption. Importantly, metrics for traditional "security" (e.g., that focus on vulnerability counts) are not necessarily applicable here—the specific focus on scientific computing integrity is narrower and much more specific.

Each individual topic, technique, and milestone will have its own additional set of metrics, however. For example, metrics for isolation within trustworthy supercomputing need to account for heterogeneous hardware and must scale well beyond traditional node-based techniques to extreme-scale systems. While such isolation metrics may include assessment of specific vulnerabilities, the metrics applied need to consider the effect of vulnerabilities on the users and the science they produce.

Similarly, metrics for success for automated verification and testing largely revolve around the ability to identify meaningful single points of failure and flaws in software and workflow implementation leading specifically to a failure of scientific computing integrity, without a prohibitively proportional increase in findings of false positives of such flaws. Additional metrics required likely include the degree of independence of redundant software components. When combined with a fault tree analysis, this can indicate the degree of reduced vulnerability to common mode failures that can occur even in the presence of redundant components that can vote against outliers in a consensus and/or accommodate failovers.

Metrics for success for vulnerability classification largely revolve around the ability to identify meaningful software vulnerabilities leading to a failure of scientific computing integrity, without a prohibitively proportional increase in findings of false positives of such vulnerabilities. Design flaws may suggest required changes to the architecture, hardware, or protocols to ensure integrity of operations.

For provenance log collection, the ability to reproduce scientific computing results within certain parameters specified by domain scientists is another key metric. Metrics for success for extreme-scale provenance analysis are primarily successful in terms of their efficacy for assuring or identifying failures and determining their source, while performing at near-real time.

Simulation and modeling metrics are largely defined in terms of how accurate simulation results compare to actual data, as collected and analyzed via the other technical areas.

Metrics for success for research on trust within high-end, open data centers largely center around the degree to which changes developed via the *Trustworthy Supercomputing and Extreme-Scale Data Analytics* topics that are usable by both domain scientists and facility system administrators can be built and deployed,. Success for research in data centers will also depend on the results of key stakeholders using those tools to validate scientific computing integrity, or where it has failed, and the ability to determine what the sources of those failures are.

Much as with the *Trustworthy Supercomputing* topic area, metrics for success of research on trust within open, high-end networks largely depend on the ability of modifications of networking protocols and their implementations to generate provenance data that helps to identify sources of loss of scientific computing integrity, and the places where those elements help to assure scientific computing integrity to avoid integrity loss.

Finally, adversarial or not, there is also a need for an element of "ground truth" for testing scientific computing integrity solutions within extreme-scale computing environments. To gain this, "red team" / "blue team" exercises using the flaw hypothesis methodology might provide an initial set of data to test against while also providing DOE with a clearer sense of the vulnerabilities that exist in our current HPC systems, and a set of benchmark data for program evaluation.

6. Conclusion

This workshop report has described a potential pathway for the research and development needed in order to support science- and engineering-based solutions to scientific computing integrity and overall computer security for high-performance computing systems, as scientific results increasingly rely on fabulously complex computing systems to collect, transmit, store, and analyze data, and to model and simulate aspects of the natural world. This report could serve as a starting point to inform integrity technologies within the context of the Exascale Computing Initiative (ECI); however, our attempt is to look beyond ECI.

As stated in our previous report [1], by supporting strong research and development now, ASCR will enable the foundational basis for assuring extreme-scale scientific computing integrity as it moves well into the 21st century, continuing its leadership, heritage, and legacy of large-scale high-performance computing, while also discovering and developing techniques that will undoubtedly have broad application.

References

[1] S. Peisert, *et al,* "ASCR Cybersecurity for Scientific Computing Integrity," Technical Report LBNL-6953E, U.S. Department of Energy Office of Science report, February 2015.

[2] R. P. Feynman, *The Presidential Commission on the Space Shuttle Challenger Accident Report*, Volume 2, Appendix F: "Personal Observations on the Reliability of the Shuttle," June 6, 1986.

[3] N. Leveson (editor), *An Assessment of Space Shuttle Flight Software Development Processes*, National Academies Press, Washington, D.C., 1993.

[4] L. J. Osterweil, "Software Processes Are Software Too," *Proceedings of the 9th International Conference on Software Engineering*, pages 2–13, 1987.

[5] R. J. Anderson, *Security Engineering: A Guide to Building Dependable Distributed Systems,* second edition, Wiley Publishing, 2008.

[6] M. Bishop, *Computer Security: Art and Science*, Addison-Wesley Professional, Boston, MA, 2003.[7]
F. B. Schneider (editor), *Trust in Cyberspace*, National Academies Press, 1999, Washington, D.C., 1999.

[8] S. Goodman and H. Lin (editors), *Toward a Safer and More Secure Cyberspace*, National Academies Press, Washington, DC, 2007.

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

[10] C. Stoll. Stalking the Wily Hacker, *Communications of the ACM*, 31(5):484–497, May 1988.

[11] M. Bishop, "UNIX Security in a Supercomputing Environment," *Proceedings of the 1989 ACM/IEEE Conference on Supercomputing*, pages 693–698, Nov. 1989.

[12] D. Geer, "Resolved: the Internet is No Place for Critical Infrastructure," *Communications of the ACM*, 56(6):48–53, June 2013.

[13] K. Thompson, "Reflections on Trusting Trust," *Communications of the ACM*, 27(8):761–763, August 1984.

[14] E. Organick, *The Multics System: An Examination of Its Structure*, MIT Press, Boston, MA, USA, 1972.

[15] M. D. Schroeder and J. H. Saltzer, "A Hardware Architecture for Implementing Protection Rings," *Communications of the ACM*, 15(3):157–170, 1972.

[16] P.A. Karger and R. R. Schell, "Thirty Years Later: Lessons from the Multics Security Evaluation," *Proceedings of the 2002 Annual Computer Security Applications Conference (ACSAC)*, pages 119–126, 2002.

[17] M. D. Schroeder, "Engineering a Security Kernel for Multics," *Proceedings of the Fifth ACM Symposium on Operating Systems Principles (SOSP)*, pages 25–32, 1975.

[18] F. P. Brooks, "No Silver Bullet —Essence and Accidents of Software Engineering," *IEEE Computer*, 20(4):10–19, April 1987.

[19] B. Boehm, "A View of 20th and 21st Century Software Engineering," *Proceedings of the 28th International Conference on Software Engineering (ICSE)*, pages 12–29, 2006.

[20] M. Schaefer, "If A1 is the Answer, What Was the Question? An Edgy Naïf's Retrospective on Promulgating the Trusted Computer Systems Evaluation Criteria," *Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC)*, pages 204–228, 2004.

[21] B. Snow, "We Need Assurance!" *Proceedings of the 21st Annual Computer Security Applications Conference (ACSAC)*, 2005.

[22] H. D. Mills, M. Dyer, and R. C. Linger, "Cleanroom Software Engineering," *IEEE Software*, 4(5):19–25, 1987.

[23] B. Blakley. "Assuring Critical Systems," *Proceedings of the 2002 New Security Paradigms Workshop (NSPW)*, pages 110–112, 2002.

[24] P. Karger, M. E. Zurko, D. W Bonin, A. H. Mason, and C. E. Kahn, "A Retrospective on the VAX VMM Security Kernel," *IEEE Transactions on Software Engineering*, 17(11):1147–1165, 1991.

[25] G. Klein, J. Andronick, K. Elphinstone, T. Murray, T. Sewell, R. Kolanski, and G. Heiser, "Comprehensive Formal Verification of an OS Microkernel," *ACM Transactions on Computing Systems*, 32(1):2:1–2:70, February 2014.

[26] C. E Landwehr, "A Building Code for Building Code: Putting What We Know Works to Work," *Proceedings of the 29th Annual Computer Security Applications Conference (ACSAC)*, pages 139–147. ACM, 2013.

[27] J.A. Ang, *et al*., "Abstract Machine Models and Proxy Architectures for Exascale Computing," U.S. Department of Energy Office of Science report, May 2014.

[28] U.S. Department of Energy Office of Science and National Nuclear Security Administration, "Preliminary Conceptual Design for an Exascale Computing Initiative," November 2014.

[29] M. Barry, *et al*., Machine Learning and Under- standing for Intelligent Extreme-Scale Scientific Computing and Discovery, U.S. Department of Energy Office of Science report, July 2015.

[30] K. Asanović, R. Bodik, B. C. Catanzaro, J. J. Gebis, P. Husbands, K. Keutzer, D. A. Patterson, W. L. Plishker, J. Shalf, S. W. Williams, and K. A. Yelick. "The Landscape of Parallel Computing Research: a View from Berkeley," Electrical Engineering and Computer Sciences, University of California at Berkeley, Tech Rep. No. UCB/EECS-2006-183, December 2006.

[31] V. R Basili, "Language as a Tool for Scientific Programming," *Proceedings of the Third ICASE Conference on Scientific Programming*. Academic Press, 1976.

[32] D. E. Denning, "The Limits of Formal Security Models," National Computer Systems Security Award Acceptance Speech, http://www.cs.georgetown.edu/denning/infosec/award.html, Oct 1999.

[33] Inder Monga, *et al*., Operationalization of Software-Defined Networks (SDN) Program Review, December 2013.

Appendix: Workshop Organizing Committee

Sean Peisert, (Workshop Chair and Trustworthy Supercomputing Topic Lead), Lawrence Berkeley National Laboratory and University of California, Davis

Thomas Potok, (Knowledge and Analytics Topic Lead), Oak Ridge National Laboratory

Todd Jones, (Networks and Data Centers Topic Lead), Sandia National Laboratories

Appendix: DOE ASCR Point of Contact

Robinson Pino, U.S. Department of Energy

Appendix: Workshop Participants and Other Contributors

Ryan Adamson,	Oak Ridge National Laboratory
Erik Boman,	Sandia National Laboratories
Aydın Buluç,	Lawrence Berkeley National Laboratory
Nick Buraglio,	Energy Sciences Network (ESnet)
Franck Cappello,	Argonne National Laboratory
Stephen Crago,	University of Southern California/ ISI
Jonathan Crussell,	Sandia National Laboratories
Brent Draney,	National Energy Research Scientific Computing Center (NERSC)
Mike Fisk,	Los Alamos National Laboratory
Ian Foster,	Argonne National Laboratory
Brian Gaines,	Sandia National Laboratories
Daniel Gunter,	Lawrence Berkeley National Laboratory
Aric Hagberg,	Los Alamos National Laboratory
Mahantesh Halappanavar,	Pacific Northwest National Laboratory
William Hart,	Sandia National Laboratories
Steven Hofmeyr,	Lawrence Berkeley National Laboratory
Craig Jackson,	Indiana University
Sushil Jajodia,	George Mason University
Todd Jones,	Sandia National Laboratories
Sven Leyffer,	Argonne National Laboratory
Joshua Letchford,	Sandia National Laboratories
Vitus Leung,	Sandia National Laboratories
Cong Liu,	University of Texas, Dallas
Robert Lucas,	University of Southern California/ ISI
David Manz,	Pacific Northwest National Laboratory
Barton Miller,	University of Wisconsin, Madison LeAnn Miller, Sandia National Laboratories
Jose Nazario,	Soar Technology, Inc.
Sean Peisert,	Lawrence Berkeley National Laboratory
Thomas Potok,	Oak Ridge National Laboratory
Stacy Prowell,	Oak Ridge National Laboratory
Siva Rajamanickam,	Sandia National Laboratories
David Robinson,	Sandia National Laboratories

Marcus Schordan,	Lawrence Livermore National Laboratory
Michael Sinatra,	Energy Sciences Network (ESnet)
Adam Sypniewski,	Soar Technology, Inc.
Trung Tran,	Altera Corporation
Jian Yin,	Pacific Northwest National Laboratory
Steven Young,	Oak Ridge National Laboratory
Peter Zbiegiel,	Argonne National Laboratory

Appendix: Workshop Observers

Barbara Helland,	U.S. Department of Energy
Mark Thornock,	U.S. Department of Energy
Peter Tseronis,	U.S. Department of Energy
Walter Dykas,	U.S. Department of Energy
Lucy Nowell,	U.S. Department of Energy
Larry Lanes,	U.S. Department of Energy

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.