Towards a Comprehensive Mapping of the Extreme-Scale Software Life-Cycle and Its Implications for Security

Barton P. Miller*, Franck Cappello**, Miron Livny*
*University of Wisconsin, **Argonne National Laboratory

Focus area: Trustworthy Supercomputing

Can we trust that the results of computation are not affected by malicious tampering?

This question covers a much broader scope than just the integrity of numerical simulation and data analytics executions. It refers to the problem of protecting the **end-to-end** workflow, from the moment of first coding and acquisition of input data to the delivery of results to decision makers. This end-to-end model must include all the steps of computation, communication and storage.

This question can be decomposed into three parts:

The first part is to understand the workflow building, deploying, running and analyzing results of numerical simulation and data analytics applications. Where these components are coming from (provenance)? What is the life cycle of each component (data, software and hardware)? How are these components combined together? How reproducible is the result? We need to protect or validate the integrity of (1) the various stored data (input to computations, intermediate results stored, and final results) [data at rest], (2) the computations (running programs) from being manipulated [data in progress], and (3) the movement of the data [data in transit]?

The second part is to identify threats and risks at each step of the workflow. Threats are numerous coming from software and hardware components as well as from deployment. While there has been good research done on identifying issues that relate to the security of numerical simulation and data analytics executions, the community is clearly lacking a threat model[1] covering the different steps of the workflow, against which mitigation approaches could respond effectively and efficiently.

The third part is the definition of trust models, where components involved in the data manipulation and transformation are linked together through a graph of trust relations. Trust relations depend on how the user build trust on the execution results, on the provenance of the execution components (data, hardware and software), on their reputation built from user experiences and verification (formal when possible) [Cap2015].

To address these questions, we are proposing a process where we precisely characterize the workflow used in the development and deployment of an HPC computation, laying out steps of the workflow in detail, describing the principals participating at each step, the threats and risks present and the trust relations between its components (data, hardware and software components, deployment).

[1] To keep the analogy with fault model in the resilience domain.

Our goal is to help insure that executions of critical DOE simulation and analytics workloads provide timely and intended results.

To this end, the workflow approach would server several purposes:

1. Provides an overview of what activities could interfere with providing timely and intended computational results. By grouping the reporting of the threats, we can identify the most common and imminent ones. By having an overview of the risks, we can determine which ones have the most serious impact and prioritize the remediation. The result is an effective prioritization of effort and resources.
2. Describes each step of the software development and deployment process, showing how to address the threats that come up at each step. By describing the threats at each step, we can address them individually, allowing the work to be spread across multiple teams.
3. Describes, in addition to the risks and threats at each step, the actors involved, inside members of the development, management or operational teams, or outside attackers.
4. Represents a reference from which any deviation in the data source, software development or deployment would be detected and would need to be assessed.
5. Provides a reference to be used to compare to the assurance of other software development and deployment scenarios (such as e-commerce sites or financial services). The goal is to leverage best and most effective practices from other disciplines, accelerating the development of assurance solutions and identifying unique areas in which the HPC community must find their own solutions.
6. Provides an attack plan for assurance teams whose job it is to test the effectiveness of the strategies being used to validate the security of the HPC software development and deployment process. The testing phase of the assurance effort is as important as the design and implementation phases.

This approach opens research problems that would need to be covered to provide a well-defined process for workflow construction and monitoring:

A. The integrity of the storage, computation and communication need to be characterized, and the impact of breaches in such integrity need to be evaluated and their risk needs to be quantified.
B. A workflow description method that includes both the computational environment and the participants. These participants are a multi-disciplinary group that includes scientists, software developers, testing/QA groups, security, and assurance groups.
C. A workflow monitoring environment need to be developed based on operational, monitoring efficiency and detection effectiveness constraints.
D. Tools need to be developed to monitor integrity and assess the correctness and completeness of the workflow.
E. Identifying areas of overlap between those techniques that deal with malicious faults and those that deal with component failures and programming errors. Tracking the provenance is an important example of such overlap.

[Cap2015] F. Cappello et al., “Improving the trust in results of numerical simulations and scientific data analytics”, ANL tech report ANL/MCS-TM-352, Argonne National Laboratory, 2015