GRISL : Protecting and Hardening Scientific Use Of Software Libraries With GRISL

PI: Justin Cappos (New York University) Co-PI: Vidya Lakshmi Rajagopalan (New York University)

https://github.com/Lind-Project

Project Overview

Scientific workflows increasingly rely on highly optimized C/C++ libraries, which often contain vulnerabilities and bugs.

- **Harden unsafe libraries** without changing the scientists' code.
- Use lightweight user space isolation to prevent crashes and data corruption.
- Deliver pre-hardened libraries with ~1% performance overhead.
- Improve reliability of HPC, AI/ML, and scientific workflows nationwide.

Cyber Security Innovation Being Transitioned

High performance, backwards-compatible, legacy library isolation for HPC utilizing WebAssembly (WASM) sandboxing principles.

- **Library Isolation:** Established software compartments called *cages* to ensure that a single library failure cannot compromise the broader application ecosystem.
- **Dynamic Loading:** Enabled dynamic library loading in WASM, allowing for the secure isolation of individual libraries into distinct compartments.
- **Support for HPC applications:** Engineered an API for local, *inter-cage*, and remote library calls to enable seamless execution of distributed applications.
- **Policy Interposition:** Created customized, per-library system call reference monitors that enable tailored OS restrictions per library.

Approach For Transitioning the Innovation

- **Strong community relationships:** Build on 15+ years of secure software deployments (TUF, Uptane, in-toto, etc.)
- Open-source release with early / frequent community engagement (SKACH, etc .) and strategic outreach via ORCA under Linux Foundation to gather early feedback and accelerate adoption.

Benefits to Scientific Cyberinfrastructure

Society needs science to be accurate, but libraries (AI/ML) often are performance-optimized and have hidden flaws. Isolating these libraries ensures:

- A significant reduction in crashes and silent data corruption in scientific workflows.
- Improved reproducibility and reliability of computational research results.

Transition Evaluation & Demonstration Plan

Metrics for success:

- Substantial improvements to ~10 widely used libraries.
- Minimal performance overhead (~1% target).
- Reduction in crashes, data corruption, and debugging time reported by users.

Integration into major HPC and cloud research platforms (JupyterHub, Open OnDemand, CloudLab, Chameleon).

Open-source implementation and community: Prebuilt packages available via Conda, pip, and Spack for easy installation.

Risks Versus Potential For Advances

- Risks: Integrating isolation into complex, performance-critical libraries may expose unforeseen critical compatibility / performance issues. Adoption requires sufficient POC / momentum.
- Payoff: Increased reliability, safety, and reproducibility for HPC, AI/ML, and scientific computing. This framework can be applied to AI, etc. use across domains.