CICI: UCSS: Secure Containers in High-Performance Computing Infrastructure

PI: Yuede Ji (University of Texas at Arlington), Co-PI: Xing Gao (University of Delaware)

Goal: ***Designing secure container solutions for high-performance computing (HPC) infrastructures.***

Motivation:

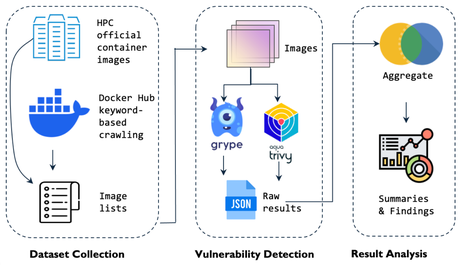
- ***Container has become prevalent in major HPC infrastructures***, e.g., Summit, Sierra, Piz Daint, and Frontera.
- ***Container images are insecure***. For example, a recent study on neuroscience container images shows that there are *460 vulnerabilities per image*.
- ***The weak isolation between containers and hosts can lead to vulnerabilities.*** We have observed 11 such vulnerabilities in our study since 2017.

Project 1: Large-scale container image security investigation in HPC environment

- Collected 4,784 container images in total, including 3,061 HPC infrastructure images and 1,723 HPC applications images.
- Combine multiple tools to detect vulnerabilities across multiple layers of the software

Findings:

- The container images used in HPC have a large number of vulnerabilities, i.e., 2,109 vulnerabilities per image.
- The detected vulnerabilities tend to be highly serious
- Most vulnerabilities have very low likelihood of being exploited.
- Certain vulnerabilities with very high EPSS scores.
- Half of the vulnerabilities already have their patches available



Thrust 1: Efficient vulnerability detection for container images

- ***Goal***: Designing *an efficient image vulnerability scanner* to detect the images uploaded to the HPC infrastructure.
- Task 1-1 converts different types of code to embedded control flow graph (ECG).
- Task 1-2 converts ECG to code embedding with graph neural network and triplet-loss network.
- Task 1-3 proposes an efficient locality-sensitive hashing-based online vulnerability searching method.

Project 2: Context-aware exploitability assessment

Motivation:

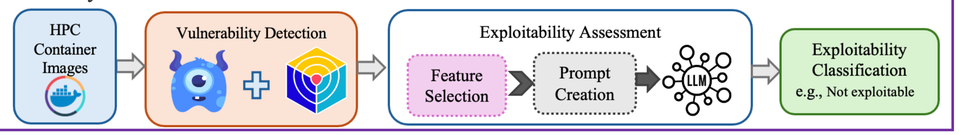
- Thousands of CVEs ≠ Exploitable; CVSS ≠ real-world risk; Exploitation depends on environment constraints

Our Methodology:

- Model exploitability as a context-dependent property shaped by technical feasibility, environmental exposure, operational constraints, and empirical attacker behavior
- Utilize multiple large-language models with structured prompts to synthesize exploitability evidence

Findings:

- Container-aware prompting consistently suppresses high exploitability predictions across models, showing that deployment-aware prompting improves exploitability assessment
- Deployment-aware signals, such as package presence, remediation status, and applicability constraints, strongly shape exploitability assessments.



Thrust 2: Secure, lightweight and high-performance container runtime

- ***Goal***: Designing *a container runtime tailored for the HPC infrastructure*, which is both secure and high-performance.
- Task 2-1 uses a lightweight virtual machine hypervisor as the container runtime with various optimizations.
- Task 2-2 customizes the runtime based on HPC requirements on hypervisor feature, file system, network, and GPU.
- Task 2-3 designs a dynamical image debloating method that can remove unnecessary files, software, and packages.

Project 3: Secure HPC Container Runtime Development

Approach:

- Using lightweight virtual machine based on Rust to achieve container-like latency while preserving isolation
- Pruned hypervisor with only necessary modules to create VMs to reduce software complexity
- Customized image with minimum required packages that supports wide-range of HPC workloads
- Workload scheduler to improve performance by leveraging hardware features

Progress and on-going efforts:

- Designed and implemented container runtime supporting HPC workloads
- Working on managing CPU features such as simultaneous multithreading (SMT) and non-uniform memory access (NUMA), as well as GPU features, such as NVIDIA Multi-Instance GPU (MIG).
- Considering GPU L3 TLB and PCIe contentions on designing workload scheduler