NISTIR 8297

Interoperability of Web Computational Plugins for Large Microscopy Image Analyses

Peter Bajcsy
Software and Systems Division
Information Technology Laboratory
National Institute of Standards and Technology

Nathan Hotaling
National Center for Advancing Translational Sciences
National Institutes of Health

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8297

March 2020

U.S. Department of Commerce
Wilbur L. Ross, Jr., Secretary

National Institute of Standards and Technology
Walter Copan, NIST Director and Undersecretary of Commerce for Standards and Technology

National Institute of Standards and Technology Interagency or Internal Report 8297
33 pages (March 2020)

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8297

Certain commercial entities, equipment, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by NIST, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

There may be references in this publication to other publications currently under development by NIST in accordance with its assigned statutory responsibilities. The information in this publication, including concepts and methodologies, may be used by federal agencies even before the completion of such companion publications. Thus, until each publication is completed, current requirements, guidelines, and procedures, where they exist, remain operative. For planning and transition purposes, federal agencies may wish to closely follow the development of these new publications by NIST.

- Summary: For open source algorithmic plugins, registries for Docker images can become "trusted" by putting all code to GitHub for informal reviews, having mechanisms for validating plugins, and by restricting access to registries for Docker images.

2. How hard is it to scan plugins? What tools exist?
   - Discussion Inputs:
     - Docker has some tools for container scanning.
     - Even non-malicious code can cause problems by unintentionally misbehaving.
     - Services such as "the Snyk Security Scan " could be useful. "The Snyk Security Scan Azure Pipelines Task scans your application dependencies and container images for open source security vulnerabilities."
     - Awareness of security is important. May require a cultural change.
   - Discussion Opinions:
     - Can you trust certifications? Can you identify who certified it?
     - Will security updates break reproducibility?
     - Do plugins have network access? May need to access large external datasets.
     - Open-source containers are more trustworthy. Closed-source containers require trusting the author.
     - Many organizations feel they do not have resources to guarantee security. In this case:
       - Make "best effort".
       - Rely on the threat of punishment to prevent malicious behavior.
   - Summary: More research and in-depth discussions are needed to understand how to design security scanning tools for Docker images and Docker containers, and how to combine the tools in continuous integration/continuous delivery (CI/CD) workflow. Commercial vendors are leading this effort to meet Security Technical Implementation Guides (STIGs) [13] and those efforts can be leveraged in scientific workflows.

3. Who should be responsible for security of Docker container execution? Plugin author? Plugin repository manager? User? How to deliver secure plugins?
   - Discussion Opinions:
     - Users don't want to spend the time and effort to scan, but they are the group who are ultimately responsible for security, since they are the ones who will be impacted by vulnerabilities.
     - Security requirements depend on the identity and resources of those who you expect would want to attack you (your threat model).
     - Plugin author should sanitize inputs.
     - Would ranking/reviews for plugins become an incentive to make plugins secure?
     - Plugin authors should sign containers. Signatures need to be checked.
     - How do you determine what signatures to accept? Manually validate identities by directly contacting the signers?
     - Should communication between plugins be required to be encrypted? HTTPS should generally be required.
     - Security is a lot of work. One should use Docker security features, best-practices, and namespaces (filesystems and privileges) [15]
   - Summary: Security of workflow execution consisting of chained Docker containers is the responsibility of users selecting containerized algorithmic plugins and system administrators managing the software and hardware resources running the workflow. A consensus is possible to address this security aspect by plugin authors signing a Docker image representing his/her algorithmic plugin.

This publication is available free of charge from: https://doi.org/10.6028/NIST.IR.8297