DYKONDO: Debloating Container Images for Reduced Attack Surface and Edge Deployments

Jonathan Dorn, Zachary Fry, and Adam Seitz (GrammaTech, Inc.)
https://www.grammatech.com/ research@grammatech.com

Containerization is a standard method for enhancing portability and reliability in commercial and government deployments. However, container images often balloon to hundreds of megabytes or gigabytes in size, resulting in a large attack surface and onerous resource requirements to run. Automatically debloating images reduces attack surface and resource requirements to enable deployment at the edge.

Motivation and State of the Art

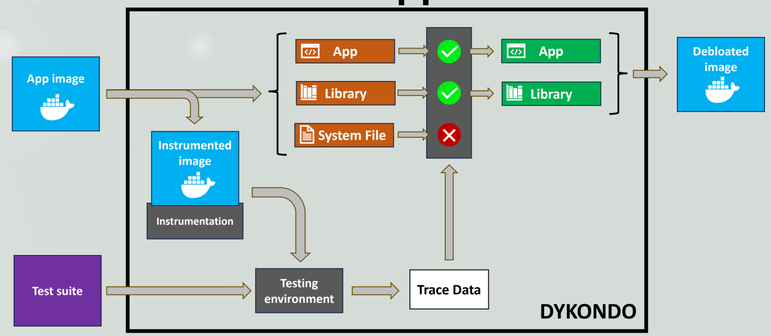
What's wrong with containers?

- Bloat results in containers that are often hundreds of megabytes or gigabytes in size.
- Large containers result in a large attack surface and impact resource requirements such as bandwidth for deployment and storage, which can limit their use on edge devices.

How are containers debloated today?

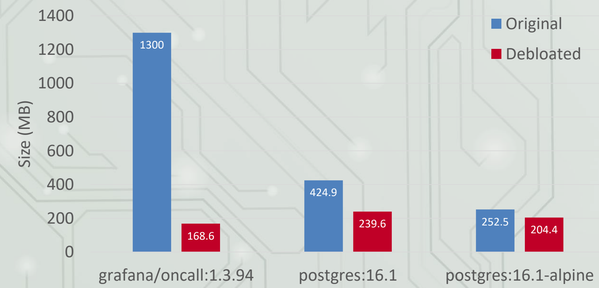
- Developers manually rewrite Dockerfiles to generate small images.
- Manual approaches may be incomplete or inconsistent and risk removing files that are required by the application.

Our Approach



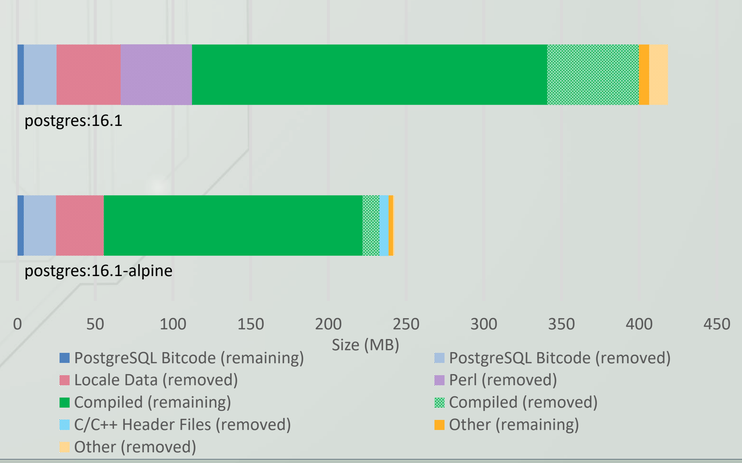
We debloat by tracing which files are accessed while running tests, unpacking the image, removing unused files, and then packing remaining files to create a new image.

Debloating Results



- 20%-87% size reduction in open-source case studies
- Alpine-based postgres, even with a slim base image, is reduced by 20%

Remaining/Removed Files



Future Work

- Quantify impact on container security
- Integrate binary hardening and debloating using GrammaTech's GTIRB rewriting tools (https://grammatech.github.io/prj/gtirb/)
- Language-specific debloating (Python, Node.js, etc.)
- Investigate better debloating specifications:
  - Enhancing or extending test suites
  - Using non-executable specifications
- Debloat alternative targets beyond Linux containers, such as VMs, Windows containers, or firmware

GRAMMATECH

This material is based upon work supported by the Office of Naval Research (ONR) under Contract No. N00014-21-C-1032. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the ONR.

THE TWENTY-FOURTH ANNUAL
HIGH CONFIDENCE SOFTWARE AND SYSTEMS CONFERENCE
MAY 6 - 8, 2024 | http://cps-hcss.org