Automatic Security Patching for Containerized Java Applications

Olufogorehan Tunde-Onadele
oatundeo@ncsu.edu
North Carolina State University
Raleigh, North Carolina

Xiaohui Gu
xgu@ncsu.edu
North Carolina State University
Raleigh, North Carolina

ABSTRACT
Containers have become increasingly popular in distributed computing environments. However, recent studies have shown that containerized applications are susceptible to various security attacks. We need to address underlying software vulnerabilities in these applications. Existing research has attempted to analyze the underlying software bugs with various static and dynamic approaches to extract vulnerability patterns in the application. However, these patterns can vary among applications and libraries. In addition, many solutions do not address the operational constraints and objectives of distributed infrastructures. In this work, we explore static program analysis to find and understand unique code patterns in the core Java libraries that are vulnerable to security attacks. Thereafter, we study how to deliver appropriate patches to Java applications in distributed container clusters.

CCS CONCEPTS
• Security and privacy → Virtualization and security.

KEYWORDS
Container Security, Program Analysis, Security Patching

1 INTRODUCTION
Containers have become increasingly popular in distributed computing environments because they provide an efficient and lightweight deployment method for various applications. These advantages have made containers the subject of recent research. However, studies [3][5] have shown that containers are prone to various security attacks, which has become one of the top concerns for users to fully adopt container technology [1].

Containerized applications also pose a set of new security challenges to distributed computing environments such as conventional computing clouds and data centers that use virtual machines. First, container image repositories are prone to vulnerabilities. A previous study [5] reveals an alarming degree of vulnerability exposure and spread in the official Docker Hub container repository. It is complex to maintain a public or private container repository which often consists of a large number of container images and many inheritance layers. If a container is created from a base image, any vulnerability detected in the base image needs to be patched in the containers that are built on top of the base image. Second, containers are often allocated with limited resources because a large number of containers often share the resources of a single physical host. Security patching causes significant resource increase (e.g., memory bloating) in a patched container, which could make the container unable to run after patching.

In addition, existing software repair fixes underlying software bugs using various static and dynamic approaches to extract vulnerability patterns in the application [2, 4, 6–8]. However, if these patterns are based on application code external to the core libraries of the programming language, the system may not detect similar issues in other applications and libraries because of differences among their code structures.

In this work, we explore static program analysis to find and understand unique code patterns in the core Java libraries that are vulnerable to security attacks. Since applications that build on these key libraries will eventually call the underlying library functions, such analysis can apply to diverse applications. After we extract such patterns, we study how to deliver suitable patches to Java applications in distributed container clusters. We conduct this study on real world security bugs in popular Java server applications.

REFERENCES