Code Reuse Attacks in Memory-Managed Languages

Eric Cornelissen

Abstract

Code reuse attacks repurpose legitimate code of a program in unintended ways to perform malicious actions. Well-known examples include return- and jump-oriented programming and data-only attacks, which affect memory unsafe languages. Perhaps surprisingly, memory safe languages are subject to similar attacks, as witnessed by existing work on object injection and prototype pollution vulnerabilities. These attacks work, despite memory safety guarantees, by exploiting language design features that provide an interface for controlled navigation of the program's shared memory abstractions, for example inheritance and reflection capabilities.

I will present a conceptual framework that we developed which crosses language boundaries in order to understand code reuse in memory-managed languages. This framework enabled us to study the root causes of attacks by identifying attack primitives and their combinations in a language-agnostic way. We demonstrate the applicability of the framework by instantiating it on five known attacks and show how it can serve as a useful tool to establish taxonomies for deeper understanding of attacks and defenses. Finally, we provide a categorization of defenses against these attacks with concrete examples and suggestions for language designers and researchers.

KTH Royal Institute of Technology, ericco@kth.se